Universidad de Valladolid

FACULTAD DE CIENCIAS

TRABAJO FIN DE GRADO

Grado en Física

Redes Neuronales Convolucionales para detectar la cubierta de nubes en el cielo

Autor: Bruno Longarela Fuente
Tutores: Roberto Román Díez y Fernando Buitrago Alonso
2024

Abstract

In the field of atmospheric physics, one of the most relevant magnitudes is cloud cover, which is defined as the proportion of sky or celestial dome covered by the clouds. Clouds play a fundamental role in regulating the atmospheric energy balance and the hydrological cycle, and they are one of the pillars of atmospheric circulation, modulating parameters such as irradiance and temperature. Therefore an accurate quantification of cloud cover is of great interest in climate and atmospheric studies.

Traditionally, cloud cover measurement has been conducted manually by experienced observers, which implies a high personnel cost and a low temporal resolution. Among the technological advances in cloud cover detection, all-sky cameras stand out for their wide field of view thanks to fisheye lenses. The images obtained allow for the automation of cloud cover measurement through image processing algorithms.

The main challenge of classical algorithms lies in the detection of thin clouds and in the handling of scenarios with high aerosol concentrations under adverse weather conditions. Recent advances in image processing have enabled the development of new strategies to improve cloud cover quantification. This work proposes an approach based on sky image processing through the use of convolutional neural networks (CNNs). This methodology allows for the identification of additional features, such as textures and edges, which are crucial for a more accurate quantification of cloud cover.

For model training and validation, 71,991 images captured by seven all-sky cameras located in the province of Valladolid and the Canary Islands were used. The images were captured using a multi-exposure regime with all-sky cameras equipped with CMOS sensors. Additionally, to enhance model robustness, the dataset was augmented through the generation of synthetic images via diffusion models.

Following this methodology, two CNN models were developed, and their results were evaluated on a reserved validation dataset. The performance of both models was tested, and the impact of including synthetic images in their training was analyzed. The ResNet36 model achieved the best accuracy, reaching a level of 71.51 %. Finally, these results were compared with those obtained in previous studies, highlighting improvements both, in accuracy and mean deviation of the measurements.

This work presents a new multidisciplinary approach to the quantification of qualitative cloud cover measurements, contributing to advancements in the temporal resolution of climate monitoring and its automation.

Resumen

En el campo de la física atmosférica, una de las magnitudes más relevantes es la cobertura nubosa, que se define como la proporción de cielo o bóveda celeste cubierta por nubes. Las nubes desempeñan un papel fundamental en la regulación del balance energético terrestre y el ciclo hidrológico, además de ser un pilar en la circulación atmosférica, modulando parámetros como la irradiancia y la temperatura. Por ello, la cuantificación precisa de la cobertura nubosa es de gran interés en el estudio del clima y la atmósfera.

Tradicionalmente, la medición de la cobertura nubosa se ha realizado de forma manual mediante observadores experimentados, lo que implica un alto coste de personal y una baja resolución temporal. Entre los avances tecnológicos en la medición terrestre de nubes, destacan las cámaras todo cielo, que, gracias a sus lentes ojo de pez, capturan un amplio campo de visión de la bóveda celeste. Las imágenes obtenidas permiten automatizar la medición de la cobertura nubosa a través de algoritmos de procesamiento de imágenes.

La principal dificultad de los algoritmos clásicos reside en la detección de nubes finas y en escenarios de alta concentración de aerosoles en condiciones climáticas adversas. Los avances recientes en el procesamiento de imágenes han permitido desarrollar nuevas estrategias para mejorar la cuantificación de la cobertura nubosa. En este trabajo se propone un enfoque basado en el procesamiento de imágenes del cielo mediante redes neuronales convolucionales (CNN). Esta metodología permite extraer características adicionales, como texturas y bordes, que resultan fundamentales para una cuantificación más precisa de la cubierta nubosa.

Para el entrenamiento y validación del modelo, se han empleado 71,991 imágenes capturadas mediante siete cámaras todo cielo ubicadas en la provincia de Valladolid y en las Islas Canarias. La captura de las imágenes se ha realizado en un régimen multi-exposición mediante cámaras todo cielo equipadas con sensores CMOS. Además, para dotar de una mayor robustez a los modelos se ha aumentado el conjunto de datos mediante la generación de imágenes sintéticas por difusión.

Siguiendo esta metodología, se han desarrollado dos modelos de CNN, cuyos resultados han sido evaluados sobre un conjunto de datos reservado para validación. Se evalúa el rendimiento de ambos modelos y se analiza el impacto de la inclusión de imágenes sintéticas en su entrenamiento. El modelo ResNet36 fue el mejor alcanzando una tasa de acierto del 71.51 %. Finalmente, se comparan estos resultados con los obtenidos en trabajos previos, destacando el avance en la exactitud y en la desviación media de las medidas obtenidas.

Este trabajo aporta una nueva aproximación multidisciplinar a la cuantificación de medidas cualitativas de la cubierta nubosa, contribuyendo así a una mejora en el ámbito de la resolución temporal de la monitorización climática y a su automatización.

Índice general

Capítulo 1

Introducción

En un contexto marcado por el surgimiento de nuevos algoritmos de inteligencia artificial y la era de la información, el estudio de las nubes se convierte en un atractivo campo de aplicación. La posibilidad de utilizar herramientas avanzadas para predecir y modular el impacto meteorológico de las nubes abre nuevas vías de investigación y ofrece la oportunidad de mejorar nuestra comprensión de fenómenos climáticos complejos.

1.1. Las Nubes

De acuerdo a la definición oficial del Atlas Internacional de Nubes de la Organización Meteorológica Mundial (2021), las nubes se definen como:

"Una nube es un hidrometeoro consistente en partículas diminutas de agua líquida o hielo, o de ambas, suspendidas en la atmósfera y que en general no tocan el suelo. También pueden incluir partículas de agua líquida o hielo de mayores dimensiones, así como partículas líquidas no acuosas o partículas sólidas, procedentes, por ejemplo, de gases industriales, humo o polvo".

1.1.1. Formación de las nubes

Al contrario de la creencia popular, las nubes no están formadas por agua en estado gaseoso, sino que son estructuras formadas por agua líquida o sólida, es decir, gotas o cristales de hielo, muy pequeños, lo suficiente para mantenerse en suspensión (Forster *et al.*, 2021). Sin embargo, su formación está estrechamente ligada con la saturación del vapor de agua, lo que hace a este un elemento imprescindible para la existencia de estas. La troposfera, que da hogar a estas estructuras acuáticas, al contener un 99 % del vapor de agua atmosférico, tiene las condiciones ideales para dar hogar a las nubes gracias a su gradiente de temperatura para la saturación del vapor de agua [1] (Quirantes-Calvo & Gallego-Póveda, 2011).

La clave para entender la formación de las nubes está en la saturación, el proceso mediante el cual el vapor de agua pasa a gotas de agua o cristales de hielo en suspensión. Si tomamos el aire como una mezcla de gases (aire seco y vapor de agua) que el vapor de agua cambie de fase va a depender de las variables termodinámicas que afectan al sistema como la temperatura o la presión; así como de la fracción de vapor de agua en la mezcla, es decir, de la humedad. Cuando a una determinada temperatura el aire no sea capaz de contener más vapor de agua, diremos que está saturado [2]. A mayor temperatura, mayor cantidad de vapor puede conformar la mezcla sin

[1] Para una mayor extensión ver Apéndice A: Cuestiones sobre la formación de las nubes, La atmosfera terrestre.

[2] Para una mayor profundidad en la termodinámica de la formación de las nubes y la saturación del aire, ver Apéndice A: Cuestiones sobre la formación de las nubes, Núcleos de condensación.

que el aire llegue a saturarse y a menor temperatura lo contrario. Podemos empezar a entrever que los procesos de cambios de presión y temperatura de la troposfera son los mecanismos fundamentales para la creación de nubes. Estos mecanismos se pueden dividir en dos clases [3]:

- Mecanismos de saturación por enfriamiento: enfriamiento por ascensos (frentes, convención convergencia de viento, orografía) y enfriamiento por irradiación.
- Mecanismos de saturación por aporte de humedad: mezcla de masas de aire, convergencia de humedad y turbulencias.

1.1.2. Clasificación de las nubes

Poder clasificar las nubes es de gran utilidad para trabajar con ellas, así como conocer sus principales características. El Atlas internacional de las Nubes (2017) de la Organización Mundial Meteorológica (OMM) sigue el sistema latino *Linnaeus* de géneros y especies adoptado por Luke Howard en su libro The Climate of London (1803) para la clasificación de nubes.

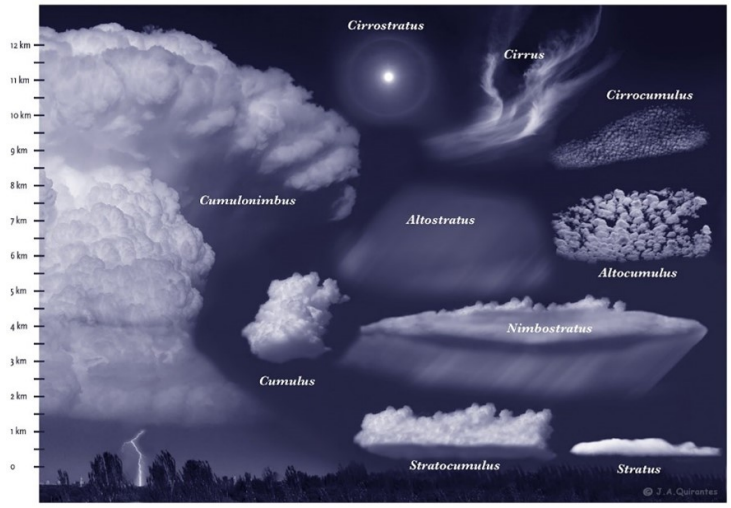

Figura 1.1: Los 10 géneros de las nubes (Quirantes-Calvo & Gallego-Poveda; 2011)

- Se reconoce 10 géneros de nubes (ver Figura 1.1): *Cirrus* (Ci), *Cirrocumulus* (Cc), *Cirrostratus* (Cs), *Altocumulus* (Ac), *Altostratus* (as), *Nimbostratus* (Ns), *Stratocumulus* (Sc), *Stratus* (St), *Cumulus* (Cu) y *Cumulonimbus* (Cb). Estos géneros son excluyentes mutuamente y conforman la característica principal de la nube subdividiéndose a su vez en especies y variedades.
- Hay 14 especies de nubes, que dependen de la forma de la nube y de su estructura interna: *fibratus, uncinus, spissatus, castellanus, floccus, stratiform, nebulosus, lenticularis, fractus, humilis, mediocris, congestus, calvus y capillatus.*
- Existen 9 variedades de nubes, refiriendose a la disposición y transparencia de estas: *intortus, vertebratus, undulatus, radiatus, lacunosus, duplicatus, translucidus, perlucidus y opacus.*
- Además se dispone de 9 rasgos distintivos y nubes accesorias: *incus, mamma, virga, praecipitatio, arcus, tuba, pileus, velum y pannus.*

[3]Para ver en más profundidad los mecanismos de saturación consultar Apéndice A: Cuestiones sobre la formación de las nubes, Mecanismos de saturación.

Además de su clasificación por géneros y por especies, la OMM también clasifica las nubes según su altitud. La altura de las nubes tiene como cota superior la tropopausa, ya que tras ella, la temperatura se mantiene constante en los primeros kilómetros de la estratosfera, para ascender progresivamente hasta la estratopausa. Debido a esta distribución de temperatura, la tropopausa actúa como limite superior de los movimientos verticales de aire y limita la región de formación de las nubes a la troposfera. Por debajo de la tropopausa se pueden definir tres pisos distintos de nubes: alto, medio y bajo. No podemos definir intervalos de clasificación por altura fijos, ya que dependiendo de la estación y de la latitud en la que nos encontremos (Tabla 1.1), los pisos de las nubes varían (Quirantes-Calvo & Gallego-Póveda; 2011; OMM, 2021).

Tabla 1.1: Clasificación de las nubes según su altura y región latitudinal (Quirantes-Calvo & Gallego-Póveda; 2011).

Nivel	Región Polar	Región Templada	Región Tropical
Nubes altas	3-8 km	3-13 km	6-18 km
Nubes medias	2-4 km	2-7 km	2-8 km
Nubes bajas	Superficie-2 km	Superficie-2 km	Superficie-2 km

En el piso alto encontramos a las nubes formadas por cristales de hielo, que son los Cirrus, los Cumulucirrus y los Cirrostratus. En el piso medio encontramos las nubes formadas por agua subfundida, es decir, a temperatura negativa, pero sin congelarse, que son los Altocumulus, los Altostratus, y los Nimbostratus. Por último, en el piso bajo se encuentran las nubes formadas por gotas de agua líquida, los Stratus y los Cumulustratus. Las nubes de desarrollo vertical pueden pertenecer a dos o los tres pisos al mismo tiempo. Estas nubes de desarrollo vertical, como ya se ha explicado en su formación, son los Cumulus y los Cumulonimbus.

La correcta clasificación de las nubes es fundamental para predecir sus movimientos, así como la cantidad de radiación solar que pasa a través de ellas (Martínez-Celda, 2021):

- Cada tipo de nube en función principalmente de su composición refleja un porcentaje diferente de radiación incidente. Los cristales de hielo y las gotas de agua tienen efectos diferentes sobre la dispersión de la radiación.

- Hay una correlación directa entre el porcentaje de luz que deja pasar una nube y su opacidad.

- La diferencia de altura entre nubes ocasiona dinámicas distintas, ya que en diferentes alturas la circulación atmosférica se comporta de manera diferente.

1.1.3. Balance radiativo y el ciclo hidrológico

La importancia de las nubes reside en su fundamental papel dentro del balance radiativo atmosférico (Boucher *et al.*, 2013), en el ciclo hidrológico y en la modulación del clima.

Las nubes poseen un alto albedo, que se define como la proporción existente entre la energía luminosa que incide en una superficie y la que se refleja. Es decir, durante el día las nubes reflejan una gran parte de la radiación solar lo que contribuye al enfriamiento global de la superficie terrestre (Forster *et al.*, 2021). Dependiendo de su tipo y densidad, las nubes pueden reflejar una mayor o menor cantidad de radiación solar. Los gases que conforman la atmosfera

son parcialmente transparentes a la radiación de onda corta (radiación solar) [4] (Quirantes-Calvo & Gallego-Póveda; 2011). La radiación solar que traspasa la atmósfera, es absorbida y remitida por la superficie terrestre en forma de radiación infrarroja o de onda larga. La radiación de onda larga emitida por la tierra es parcialmente absorbida por las nubes y reemitida de nuevo hacia la superficie. Esto se conoce como efecto invernadero, el cual no solo se da por acción de las nubes, sino, que también se da por la acción de algunos gases conocidos como gases de efecto invernadero como el CO_2 o el Metano. Su mayor efecto tiene lugar durante las noches cuando las nubes absorben la radiación infrarroja de onda larga y emiten radiación de nuevo hacia la superficie terrestre, impidiendo de esta forma descensos bruscos en la temperatura. Durante el día también se da el efecto invernadero, pero se ve compensado por la radiación solar incidente reflejada.

Las nubes con más efecto de albedo sobre la radiación solar, son las nubes bajas, como los *Stratus* y los *Cumulustratus* además de alguna nube media, como los *Nimbostratus*. Mientras que son las nubes altas, como los *Cirrus* o *Cirrustratus* las que retienen más parte de la radiación infrarroja emitida por la superficie terrestre respecto de la que solar que reflejan, contribuyendo así al calentamiento global. Las nubes regulan la temperatura enfriando y calentando el planeta, impidiendo descensos bruscos en la temperatura y teniendo en promedio un efecto refrigerante sobre el planeta (Forster *et al.* 2023). Son así un pilar de la circulación atmosférica, dando lugar a patrones de viento o distribuciones de precipitaciones en diferentes regiones (Marcos-Garrachón, 2024).

Su impacto sobre los ecosistemas y la vida terrestre no se resume solo en su control sobre el balance de temperaturas, sino, que son además las estructuras más dinámicas del ciclo hidrológico o ciclo del agua [5]. Encargándose de devolver el vapor de agua en forma de precipitaciones, de nuevo a la superficie terrestre.

Actualmente, el calentamiento global sigue siendo uno de los principales problemas que afrontamos como sociedad, y el estudio de las nubes es de vital importancia, ya que según modelos desarrollados por el Grupo Intergubernamental de Expertos sobre el Cambio Climático (IPCC), se prevé una disminución en el número de nubes bajas debido al efecto de los gases invernadero, lo que conduciría a un mayor calentamiento global (Martínez-Celda, 2021). Por todo lo mencionado anteriormente se manifiesta una extensa necesidad del estudio de las nubes basado en la adquisición masiva de datos para desarrollar modelos más precisos y con mayor capacidad de previsión y monitorización.

En este trabajo, se pretende automatizar la tarea de la monitorización de las nubes en diferentes regiones mediante imágenes todo cielo [6]. Para ello se trabajará sobre el concepto de **Cobertura Nubosa**. El Atlas de nubes y meteoros (2011) de Jose A. Quillantes y Jose A. Gallego define la "cantidad de nubes", "nubosidad" o "cobertura nubosa" como la proporción de cielo o bóveda celeste cubierta por las nubes. Se denomina como "cobertura parcial" a la región de cielo oculta por las nubes pertenecientes a un único piso, y cobertura total cuando se refiere a la suma de todas las contribuciones de coberturas parciales. Podemos tener diferentes coberturas parciales de nubes en cada piso cuya suma desemboque en una cobertura total o una cobertura parcial dependiendo de su disposición. Generalmente se expresa en "octavos de cielo cubierto", indicándose así el número de octas cubiertas de la bóveda celeste.

Las nubes delgadas, así como la estructura en capas por altura de las nubes, son los principales problemas en la medida de la cubierta nubosa mediante algoritmos clásicos. Entre las nubes

[4] Se comenta en el mecanismo de saturación por convección (Apéndice A: Cuestiones sobre la formación de las nubes, Mecanismos de saturación.)

[5] Para una mayor extensión ver Apéndice A: Cuestiones sobre la formación de las nubes, Ciclo Hidrológico.

[6] El concepto de imágenes todo cielo se introducirá más adelante en la sección 1.2.

delgadas, destacan por la complejidad de su detección, los cirros. Estas nubes poseen una baja densidad óptica, además de un índice de refracción (por ser estar conformados de cristales de hielo), que favorece la refracción, en lugar de la reflexión de la radiación en el espectro visible. Esto las dota de una apariencia tenue y traslúcida.

1.2. Equipos de medida de las nubes

En la tarea de análisis de la cubierta nubosa del cielo se ha empleado tradicionalmente a observadores meteorológicos, que toman medidas subjetivas de la cubierta nubosa desde un punto de vista terrrestre. Dependen del horizonte visible y son más precisos durante las horas de luz (González-Fernández *et al.*, 2024). Estos están confinados a un lugar específico por lo que tienen una resolución espacial muy baja. Y a pesar de que la resolución temporal de estos observadores puede llegar a no ser baja, es una tarea ardua, así como innecesaria gracias a los avances tecnológicos hoy en día.

Entre los avances tecnológicos podemos distinguir dos tipos de instrumentos de medidas de nubes: los instalados sobre la superficie terrestre y los instalados sobre satélites.

Los instrumentos de tierra, tienen un campo de visión limitado, sin embargo, ofrecen una alta precisión en las variaciones locales de parámetros atmosféricos como la irradiancia solar (Tapakis *et al.* 2013), así como, una mayor resolución temporal. Los equipos de tierra se clasifican en tres grupos:

Medidores de irradiancia: son dispositivos diseñados para medir la radiación solar que incide por unidad de superficie y tiempo ($W \cdot m^{-2}$) (Tapakis *et al.* 2013). Entre los radiómetros o medidores de irradiancia o de radiación incidente, se encuentran los piranómetros (miden intervalos de longitudes de onda específicos). Estos miden la irradiancia solar global, lo que permite estimar la cobertura y características de las nubes mediante la atenuación de la radiación solar que ocurre cuando las nubes dispersan o bloquean parte de la energía incidente. La reducción de la irradiancia medida se puede comparar con valores de referencia y correlacionarse con la densidad y tipo de nubes presentes.

Cámaras de cielo espectrales: son equipos que capturan imágenes del estado del cielo en los espectros visible, infrarrojo (IR) o ultravioleta (UV) (Tapakis *et al.* 2013). Estas pueden capturar partes del cielo específicas o pueden capturar imágenes de toda la bóveda celeste, en cuyo caso se las denomina *cámaras todo cielo*. Las *cámaras todo cielo* se verán más adelante en detalle.

- Cámaras espectrales en el espectro visible: detectan nubes basándose en las diferencias de color e intensidad entre los diferentes tipos de nubes y el cielo o la tierra.
- Cámaras espectrales infrarrojas: basan su detección en la diferencia de radiación infrarroja reflejada y emitida por las nubes con la radiación reflejada y emitida por el entorno, utilizando la temperatura de brillo [7] como medida descriptiva de la radiación.
- Cámaras espectrales ultravioleta: aunque las nubes no absorben significativamente la radiación ultravioleta, sí que la dispersan. Mediante mediciones de la dispersión y mediciones en otros rangos de absorción son capaces de detectar las nubes.

Instrumentos de Teledetección Activa: estos instrumentos emiten radiación en longitudes de onda concretas, y comparan la diferencia de la radicación emitida con la recibida. lo que nos da información adicional sobre la diferencia temporal, el ángulo y la longitud de onda (Campbell & Wynne *et al.*, 2011).

[7] La temperatura de brillo se calcula aplicando la función inversa de Planck a la radiación medida.

- LIDAR: esta tecnología hace un barrido de pulsos láseres monocromáticos.(Clothiaux *et al.*, 1998). Utilizan pulsos de diferentes longitudes de onda, midiendo el tiempo que tarda cada pulso en reflejarse y regresar al sensor, lo que permite calcular distancias con alta precisión y obtener información sobre la estructura de la atmósfera y de las características de las nubes y de los aerosoles.
- Ceilómetros: emplean láseres verticales normalmente en el infrarrojo cercano. Son muy útiles sobre todo en la medición de la altura de las nubes (Ahrens, 2015). El concepto tras ellos es el de un LIDAR de una única longitud de onda y de menor potencia, siendo por tanto, un instrumento más económico.
- Radares de nubes: emiten señales electromagnéticas (normalmente microondas) hacia la atmósfera y miden el retorno de dichas señales. Permiten detectar la posición, la velocidad y el tipo de precipitación.

Los satélites principalmente entran en juego a la hora de ganar resolución espacial. Estos permiten ganar información sobre el desarrollo y formación de las nubes sobre las cortezas continentales. El instrumento de medida principal con el que constan son los llamados *imagers*, instrumentos de multi-radiometría, que miden diferentes longitudes de onda (Weng, 2011). Arking & Childs (1985) fueron los pioneros en el uso de satélites para la detección de la radiación infrarroja de las nubes mediante radiómetros. Rossow & Schiffer (1999) hicieron uso de más de 10 años de medidas para la calibración de los radiómetros, consiguiendo entre otras cosas un avance en la detección de la radiación emitida por los *Cirrus*. Se ha seguido trabajando en esta línea: Ricardelli *et al.* (2008) utilizó el satélite Meteosat y Zhao & Girolamo (2006) hicieron uso de del satélite Terra de la NASA.

La principal desventaja de la medida por satélite es la baja resolución temporal que nos proporcionan (Barbieri *et al.*, 2017; Matos *et al.*, 2017). En busca de ganar resolución temporal, se recurre a los instrumentos de tierra. En los últimos años, entre ellos se han destacado las cámaras de todo cielo, debido a su fácil instalación, mantenimiento y bajo coste (González-Fernández *et al.*, 2024), para predicciones de tiempo en rangos de tiempo cortos (Alonso-Montesinos *et al.*, 2015; Zhen *et al.*, 2017). Por lo que se puede suplir disminución en la resolución espacial con una amplia red de cámaras. Estas cámaras son capaces de tomar fotos en periodos de tiempo cortos, lo que nos da un mapeado hemisférico de la radiancia del cielo más preciso(Martínez-Celda, 2020). Por ello en este trabajo vamos a utilizar imágenes todo cielo (tomadas con dichas cámaras), proporcionadas por el Grupo de Óptica Atmosférica de la Universidad de Valladolid (GOA-UVa) [8].

1.3. Clasificación de imágenes con Inteligencia Artificial

Los avances tecnológicos en hardware nos han conducido al surgimiento de la hoy omnipresente Inteligencia Artificial. Dentro de la Inteligencia Artificial se define otra rama, el aprendizaje automático o *machine learning*, como el conjunto de métodos y algoritmos que permiten a una máquina aprender de manera automática en base a experiencias pasadas (Bosch *et al.*, 2019). Estos algoritmos de *machine learning* se han convertido en una de las principales aplicaciones de la información hoy en día, ya que es en base a la información, que aprenden; y para ello se requiere cantidades ingentes de información.

Dentro del campo del *machine learning* surge otra subrama conocida como *Deep Learning*, en la cual se crean modelos que sean capaces de comprender conceptos abstractos en base a conceptos más sencillos. Utilizan las redes neuronales artificiales profundas ("Deep Neural Networks") como el algoritmo base para crear máquinas con esta capacidad (Marcos-Garrachón, 2024). Dentro

[8]Se puede encontrar mas información acerca del GOA-UVa en: `https://goa.uva.es/`

de las redes neuronales profundas en este trabajo trataremos con las Redes Neuronales Convolucionales, que entran dentro del marco de la Visión artificial. Estas redes se especializan en el trabajo sobre imágenes. Son capaces de aprender pequeños rasgos mediante los que luego identifican imágenes.

La visión artificial se divide clásicamente en 4 tareas: clasificación, detección, segmentación y estimación de postura (García-Pajares, 2023). La clasificación determina la categoría a la que pertenece una imagen (previamente se debe indicar cuales son las categorias posibles a los que la imagen puede pertenecer). La detección localiza y clasifica múltiples objetos dentro de una imagen; es más potente en el sentido de que dentro de una misma imagen puedes obtener diferentes objetos pertenecientes a diferentes clases. La segmentación divide una imagen en segmentos o regiones significativas. Puede ser segmentación semántica (donde cada píxel se etiqueta con una clase) o segmentación de instancias (donde se identifican objetos individuales). Por último, la estimación de postura busca determinar la posición de las articulaciones de una persona o un animal. Podemos ver ejemplos de las distintas aplicaciones de la inteligencia artificial aplicada al análisis de imágenes en la Figura 1.2 y en la Figura 1.3.

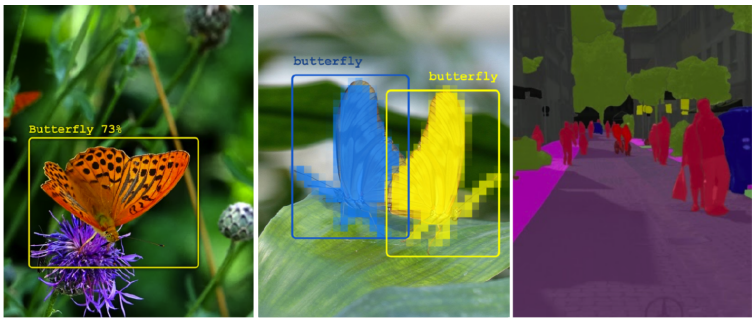

Figura 1.2: e izquierda a derecha: detección de objetos, segmentación de instancias y segmentación semántica de toda la escena (Lakshmanan *et al.*, 2024).

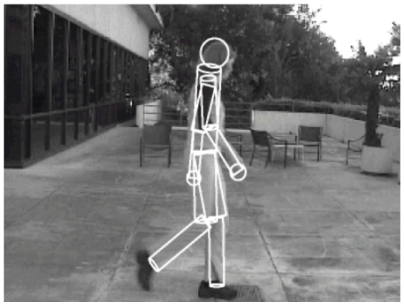

Figura 1.3: Algoritmo de estimación de postura para personas en movimiento en imagenes 2D (Sidenblah *et al.*, 2000).

Hasta la aparición de la inteligencia artificial, el análisis de nubes computacional, se realizaba mediante la segmentación de imágenes en capas dependiendo del color e intensidad del pixel. El método más extendido en este tipo de prácticas era el cociente entre los canales rojo y azul (Red Blue Ratio; RBR) ((Calbó & Sabburg, 2008; Kreuter *et al.*, 2009). Este método se basa en en las diferencias de dispersión entre el cielo despejado (dispersa en el espectro azul principalmente) y las nubes (dispersan por igual el espectro rojo y azul) (García-Pajares, 2023). Se establece así un umbral para distiguir los píxeles nube (por encima del umbral) y los píxeles de cielo despejado (por debajo del umbral) (González-Fernández *et al.*, 2024).

El método del RBR aunque sigue en uso, y funciona bien dentro de determinadas condiciones(Syazwan *et al.*, 2021; Frisch-Niggemeyer *et al.*, 2022), cuenta con limitaciones. Entre ellas podemos contar la detección de nubes finas, que no llegan al umbral (Kim *et al.*, 2016); o la alta concentración de aerosoles en condiciones meteorológicas adversas, que pueden superar el umbral, confundiéndose así con las nubes (Huo & Lu, 2009). Se han desarrollado diversos métodos para intentar dar cuenta de estos problemas. Una de las soluciones propuestas se basa en umbrales dinámicos basados en medidas estadísticas. Con ellos se ha logrado una mejor precisión en presencia de aerosoles. Algunos de estos métodos son: dependiendo del espectro azul y la zona circumsolar (Cazorla *et al.* 2008), comparando imágenes del cielo despejado simuladas con las imágenes reales de la cubierta nubosa (Ghonima *et al.*, 2012; Niu *et al.*, 2024) o considerando propiedades de simetría del cielo (Román *et al.*, 2017). Otros trabajos con umbrales adaptativos han mejorado la precisión en condiciones de neblina o nieve (superficie altamente reflectante)

(Li *et al.*, 2019).

Con el surgimiento de las nuevas Redes Neuronales Convolucionales (CNN) (Fukushima, 1980), se pretende superar los resultados en base a umbrales por color e intensidad, siendo la propia red la que aprende los factores de diferenciación entre lo que es cielo o nube, mediante el proceso de aprendizaje. Es decir, la red neuronal nos puede ayudar a discernir la cubierta nubosa del cielo. Con el desarrollo de redes neuronales convolucionales profundas entrenadas con grandes conjuntos de datos (Imagenet) (Krizhevsky *et al.* (2012)) su impacto es cada vez mayor en todas las áreas de investigación. En concreto, estas redes neuronales han probado ser de alta utilidad en el campo de las ciencias atmosféricas (Gutiérrez *et al.*, 2004), además de su eficacia en condiciones adversas, como en las de alta concentración de aerosoles en la atmósfera (Cazorla *et al.*, 2008). Eliminan la necesidad de establecer un umbral, que puede variar dependiendo de las características de la cámara utilizada (González-Fernández *et al.*, 2024). Aunque su primera aplicación mayoritariamente se centró en la segmentación de nubes (Cazorla *et al.*, 2008; Masuda *et al.* 2019; Onishi & Sugiyama, 2017; Xie *et al.*, 2020); en su uso para clasificación de imágenes, se elimina la dependencia de la clasificación pixel por pixel. La red se centra en medir la cubierta nubosa independientemente de la posición de las nubes, lo que le permite analizar características estructurales (bordes, texturas y formas geométricas) a diferencia de los métodos basados en umbrales, que solo obtienen información de la intensidad y el color. este nuevo enfoque no solo deriva en una mejora en la detección de la cubierta nubosa, además, deriva en una ventaja en la eficiencia general de la red, así como en la sencillez del modelo (González-Fernández *et al.*, 2024; Šinko, 2019).

1.4. Motivación y objetivos

Como se ha visto, las nubes tienen un gran impacto en el clima de nuestro planeta, lo que conlleva implicaciones económicas, energéticas y biológicas. Por tanto es crucial medir, monitorizar y estudiar las propiedades de las nubes. En este trabajo se pretende contribuir al analisis del impacto de las nubes mediante la cubierta nubosa. Tradicionalmente, debido a la baja eficacia de los métodos basados en umbrales, se requiere de observadores humanos que anoten la cantidad de octas de cielo ocultas por las nubes.

Como primer objetivo de este trabajo se pretende automatizar este proceso y conseguir una resolución temporal mayor en el seguimiento de la evolución de la cubierta nubosa, lo que sin duda dotaría a los modelos meteorológicos de una mayor precisión y poder predictivo a corto plazo. Para ello se adaptarán las Redes Neuronales Convolucionales para la tarea de clasificación de imágenes todo cielo en 9 clases, dependiendo del número de octas.

Otros trabajos ya se han realizado en esta línea desde la Universidad de Valladolid: Marcos-Garrachón (2024); Sanz-Huidobro (2023) (con transfer learning de U-Net); Calvo-Herrero (2023); Martínez-Celda (2021); Alegre-Fernández (2023) y González-Fernández (2024). De los cuales González-Fernández (2024) es el que cuenta con un mayor dataset de imágenes adquiriendo así una mayor precisión. Para este trabajo se ha dispuesto de un dataset ampliado con base en el utilizado por González-Fernández (2024), proporcionado por el Grupo de Óptica Atmosférica de la Universidad de Valladolid, con imágenes todo cielo.

Se establece pues, como segundo objetivo del trabajo realizar un estudio de las diferentes arquitecturas de nuestro modelo de red neuronal convolucional, analizando como varía el entrenamiento de la red y los resultados en función de los parámetros arquitectónicos, mediante los

resultados obtenidos en la validación de los modelos [9]. Se pretende investigar el uso de tranfer learning, así como investigar el uso de diferentes funciones de activación como la función Mish. Esto es posible gracias al extenso data-set proporcionado por el GOA-UVa que dota a la red de una mayor universalidad.

Previamente a explicar la metodología desarrollada para la creación y validación de los modelos, desarrollaremos dos puntos teóricos de vital importancia para que el lector, que puede estar o no familiarizado con los temas a tratar, pueda seguir la metodología aplicada:

- La instrumentación utilizada y la obtención de los conjuntos de entrenamiento, validación y test.
- Las redes neuronales, donde abarcaremos desde un breve contexto del surgimiento de las redes neuronales artificiales y como aprenden, hasta las redes neuronales convolucionales y mecanismos de optimización del entrenamiento.

[9] Un estudio más extenso sobre el entrenamiento de la red y validación del modelo, así como de los conjuntos de datos de entrenamiento y validación se hará más en adelante en los capítulos 2 y 3.

Capítulo 2

Instrumentación y conjunto de datos

2.1. *Cámaras todo cielo*

Como se ha introducido en la sección 1.2, las *cámaras todo cielo* consisten en cámaras espectrales que capturan toda la bóveda celeste. El funcionamiento de estas cámaras en la era digital se basa en los sensores CCD (*Charge-Coupled Device*) (Boyle & Smith, 1970) y CMOS (*Complementary Metal-Oxide-Semiconductor*). Ambos dispositivos se componen generalmente de una matriz de un semiconductor, dividida en fotodiodos. Al recibir la luz el semiconductor, los fotones pueden excitar sus electrones de la capa de valencia a la capa de conducción. Se crea así una corriente (dada por los electrones en la capa de conducción y los huecos en la capa de valencia) que es capturada y almacenada en forma de carga eléctrica por un circuito acoplado al fotodiodo. Cada fotodiodo representa un pixel de la imagen final. Se puede formar una imagen de esta forma, sabiendo que la carga acumulada es proporcional a la cantidad de luz que ha recibido ese fotodiodo (Antuña-Sánchez, 2021).

Los dispositivos CCD transfieren la carga acumulada entre píxeles a un amplificador central, mientras que los CMOS constan de un transistor individual para cada pixel. Aunque los CCD presentan una mejor calidad en la imagen obtenida (menos ruido), consumen más energía (Antuña-Sanchez, 2021). Esto a derivado en una tendencia en los últimos años hacia los CMOS, que gracias a los avances tecnológicos cuentan de una calidad de imagen buena, una mayor velocidad de procesado, eficiencia energética y menor costo de fabricación.

Para la identificación de colores en la imagen final, sobre el sensor se coloca un filtro óptico formado por un mosaico de celdas que siguen un patrón determinado. El más utilizado es el patrón Bayer RGGB (Bayer, 1976). Este filtro deja pasar longitudes de onda asociadas a un solo color en cada fotodiodo, es decir, en cada pixel de la imagen. Siendo el 25 % de los píxeles rojos (R), otro 25 % azul (B) y el 50 % restante verdes (G) (el ojo humano es más sensible al verde) (Antuña-Sánchez, 2021).

De esta forma, obtenemos las imágenes en formato RAW, el cual no está en color. Para obtener la interpretación en color se aplica una interpolación de color o *demosaicing*, que nos da como resultado 3 matrices correspondientes a los canales RGB (Antuña-Sánchez, 2021).

Por último, para evitar la sobresaturación de píxeles o zonas poco expuestas, se suele definir un régimen de captura multi-exposición que nos genera imágenes HDR (*High Dynamic Range*) (Antuña-Sánchez, 2021). Podemos contrastar entre sí las imágenes con diferentes exposiciones para obtener un mayor rango dinámico[1] (Debevec & Malik, 1997).

[1]El rango dinámico se define como la relación entre el tono máximo y el tono mínimo. Se puede observar una secuencia de diferentes exposiciones y la imagen final en Antuña-Sánchez (2021), pag. 51, Figura 2.13.

Las *cámaras todo cielo* utilizadas en este trabajo han sido [2]:

- **SONA202-NF**: la cámara SONA (Sistema de Observación de Nubosidad Automático) es fabricada por *Sieltec Canarias S.L.* (ver Figura 2.1 (a)). El modelo 202-NF cuenta con un sensor CMOS SONY IMX249 para un tamaño de imagen de 1172×1158 píxeles con una resolución de 2.35 mega-píxeles y 10 bits (de rango dinámico). Sobre el sensor se coloca un mosaico de filtros Bayer RGGB, y un filtro tribanda que reduce el solapamiento entre los canales de color[3] (Antuña-Sánchez, 2021). El dispositivo cuenta con una lente ojo de pez que el permite una visión periférica de 185°. El conjunto de dispositivos se encuentra encapsulado en una carcasa resistente a condiciones adversas y una cúpula de cristal transparente.

- **OMEA-3C**: la cámara OMEA es fabricada por *Alcor Systems* (ver Figura 2.1 (b)). El modelo 3C cuenta con un sensor CMOS SONY IMX178 para un tamaño de imagen de 3096×2080 píxeles con una resolución de 6.44 mega-píxeles y 14 bits. Al igual que la SONA202-NF monta un mosaico de filtros Bayer RGGB, pero a diferencia del filtro tribanda, se le ha añadido un filtro que bloquea la radiación infrarroja[4]. Además de la lente ojo de pez y las medidas de protección, este modelo cuenta con un sensor externo de temperatura y humedad conectado con un sistema calefactor en el interior de la cámara. Es por tanto este modelo una mejor opción para estaciones meteorológicas en condiciones más adversas (Antuña-Sánchez, 2021).

- **OMEA-3C-TF**: es el mismo modelo de cámara que la OMEA-3C, sin embargo monta un filtro tribanda similar al de la SONA202-NF, lo que reduce su anchura espectral (Antuña-Sánchez, 2021).

2.2. Localizaciones

Las imágenes se han extraído de 7 *cámaras todo cielo* pertenecientes al GOA-UVa dispuestas sobre el territorio español. Se tienen 2 cámaras en las Islas Canarias y 5 sobre la provincia de Valladolid, de donde provienen la mayor parte de las fotos etiquetadas. La altitud de las cámaras instaladas varía desde los *2365m* (Izaña) hasta los *630m* (La Palma de Gran Canaria). Estas condiciones nos aseguran un conjunto de datos de entrenamiento con diferencias de altitud y , con datos de regiones con diversos climas y altitudes, que dotarán a la red de una mayor capacidad generalizadora y reducirán el riesgo de sobrentrenamiento (ver Sección 3.3.2). Se pueden ver en más detalle los datos sobre cada cámara usada en el Tabla 2.1.

2.3. Conjuntos de datos

Para el entrenamiento de la red, se ha empleado un criterio humano. Es decir, distintos observadores (no necesariamente experimentados en el campo de observación de nubes, pero sí familiarizados con dicho campo) han etiquetado manualmente la cubierta nubosa del cielo en octas de las imágenes obtenidas en las diferentes localidades. El número total de imágenes etiquetadas es de 71.991, obtenidas entre enero de 2019 y diciembre de 2024. Podemos ver las el número de imágenes clasificadas correspondientes a cada cobertura nubosa (octas) en la Figura 2.1.

[2]Se puede ver una imagen de las cámaras utilizadas en la tesis doctoral de Antuña-Sánchez (2021), pag. 39, Figura 2.5

[3]Para ver el efecto del filtro tribanda sobre la respuesta espectral (transmitancia normalizada) ver Antuña-Sánchez (2021), pag. 39, Figura 2.6..

[4]Para ver el efecto del filtro que bloquea la radiación infrarroja sobre la respuesta espectral ver Antuña-Sánchez (2021), pag. 41, Figura 2.7..

Tabla 2.1: En esta tabla podemos ver el modelo de cada cámara instalado en cada localización con sus coordenadas y las fechas de toma de datos. La fecha 13/11/2024, corresponde con el fin de la adquisición de datos en aquellas cámaras que siguen en funcionamiento. Toda la información expuesta ha sido proporcionada por GOA-UVa.

ID	Cámara	Localización	Fecha inicio	Fecha fin	Latitud	Longitud	Elevación
C005	SONA202-NF	Facultad de Ciencias (Valladolid)	11/07/2018	18/05/2022	41.66	-4.71	710
C005	OMEA-3C	Facultad de Ciencias (Valladolid)	16/07/2020	27/09/2021	41.66	-4.71	710
C005	OMEA-3C	Fuencaliente (La Palma)	06/10/2021	24/01/2022	28.49	-17.85	630
C005	OMEA-3C	Izaña	04/02/2022	15/02/2023	28.31	-16.5	2365
C013	OMEA-3C-TF	Facultad de Ciencias (Valladolid)	27/06/2023	13/11/2024	41.66	-4.71	710
C014	OMEA-3C-TF	CEIP El Peral (Valladolid)	23/01/2024	13/11/2024	41.62	-4.76	692
C017	OMEA-3C-TF	ETS Arquitectura (Valladolid)	08/11/2023	13/11/2024	41.65	-4.74	705

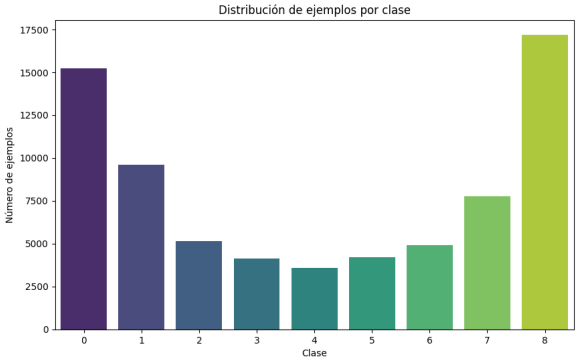

Figura 2.1: Distribución de ejemplos por cobertura nubosa (octas).

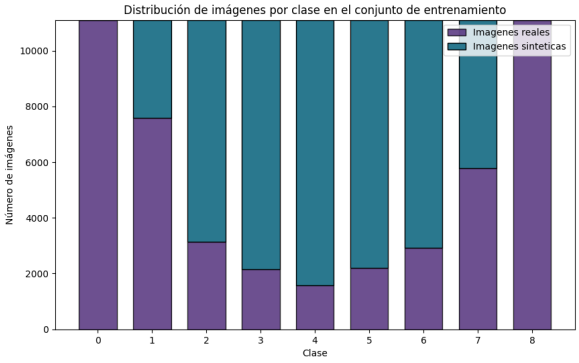

Figura 2.2: Distribución de imágenes en el conjunto de entrenamiento con división por colores entre imágenes sintéticas y reales.

A la hora de entrenar una red neuronal necesitaremos dividir los datos obtenidos en tres subconjuntos: entrenamiento, validación y test. Los datos de entrenamiento son los que se proporcionarán a la red neuronal para su optimización mediante el algoritmo de *backpropagation* (Apéndice B). El conjunto de validación (se usa junto con el conjunto de entrenamiento), no influye de forma directa en la actualización de parámetros de la red neuronal, no obstante su función es la del ajuste de hiperparámetros. Nos proporciona una forma de ver como el entrenamiento de la red actúa en tiempo real sobre un conjunto de datos reales, lo que nos permite detectar problemas en el entrenamiento. Finalmente, el conjunto de test es el conjunto sobre el que se pone a prueba la red entrenada para evaluar su precisión.

Debido a la fuerte tendencia hacia las clases en los extremos (0 octas y 8 octas), a la hora de la realización del entrenamiento se observó una fuerte dependencia de los resultados con la distribución de los ejemplos en cada conjunto. Es decir, diferentes distribuciones en los conjuntos de entrenamiento, validación y test dan lugar a un *data shift*. El modelo aprende en función de la distribución dada, y el *data shift* se puede ver como un fenómeno que no permite al modelo actuar sobre *shifts* (desplazamientos) en las distribuciones de datos. Concluimos así, que los diferentes conjuntos deben tener la misma distribución de ejemplos por clase debido a la alta sensibilidad del problema al *data shift*.

El criterio de clasificación de las clases intermedias es difuso e incluso puede variar entre observadores experimentados. Por lo que, estas serán las más difíciles de clasificar. Por tanto, dar una mayor presencia a las clases intermedias, modificando la distribución de ejemplos de tal forma que esta sea constante, mejoraría la capacidad de la red de clasificar las clases intermedias. Al

estar más presentes en cada *batch*, la red consigue aprender mejor las características de dichas clases. Sin embargo, esta metodología supondría hacer un *undersampling* en todas las clases, lo que nos llevaría a descartar un 55 % de las imágenes obtenidas (la clase con menos imágenes acota el número de imágenes que se pueden utilizar de otras clases).

Por lo mencionado anteriormente se ha seguido la siguiente metodología para armar los conjuntos de entrenamiento, validación y test:

- Se han obtenido imágenes sintéticas obtenidas mediante una red neuronal U-Net (Ronneberger *et al.*, 2015) con una metodología de difusión determinista entrenada con los datos reales de entrenamiento para cada clase [5]. Dicha red adapta los principios probabilísticos y estocásticos de las redes de difusión a un algoritmo de iteración determinista (Heitz et al., 2023). Podemos ver algunos de los ejemplos obtenidos de forma sintética en comparación con las imágenes reales en la Figura 2.3.
- Se han seleccionado 1000 imágenes reales de forma aleatoria de cada clase para el conjunto de test y otras 1000 por clase para validación.
- El conjunto de entrenamiento se ha armado con el resto de imágenes reales a las que se le ha añadido las sintéticas. Se ha hecho un *undersampling* de cada clase hasta obtener 11100 imágenes por clase.

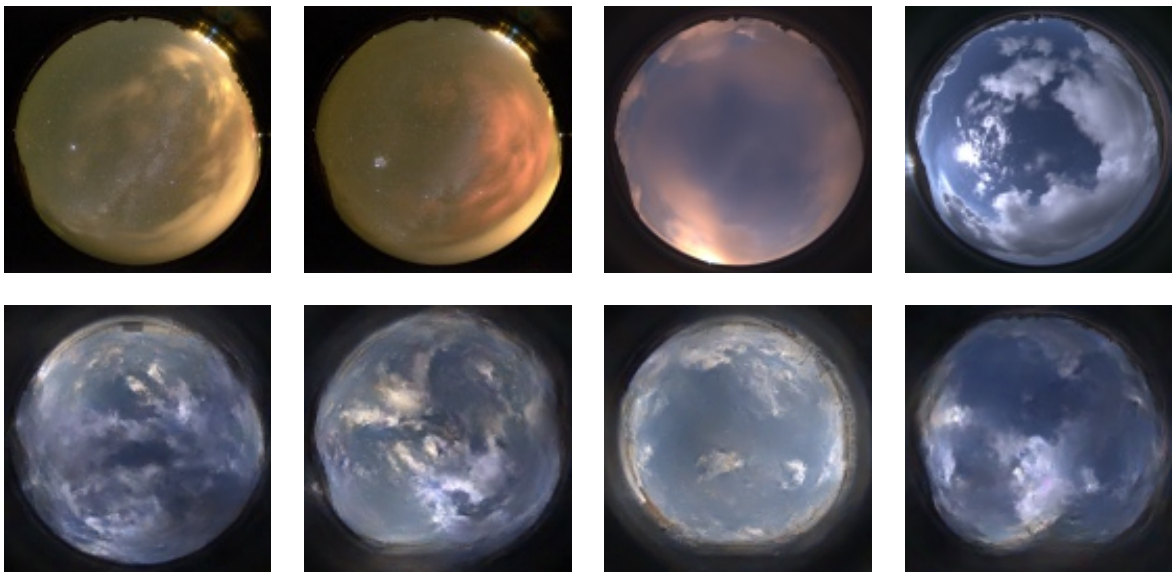

Figura 2.3: En esta figura podemos observar 4 imágenes reales sobre la fila superior, obtenidas en diferentes condiciones de iluminación mediante *cámaras todo cielo*. Las 4 imágenes que se observan en la fila inferior son imágenes sintéticas generadas mediante el algoritmo de difusión determinista de Heitz *et al.*, 2023.

Se obtienen así un conjunto de test y validación cada uno con 9000 imágenes reales; y un conjunto de entrenamiento conformado por imágenes tanto reales como sintéticas con un total de 99900 imágenes. Se puede ver la distribución de imágenes por clase en el conjunto de entrenamiento en la Figura 2.2. Conseguimos de esta forma suplir, la falta proporcional de imágenes en

[5] La metodología de difusión empleada queda fuera del alcance de este trabajo. Para la realización de las imágenes sintéticas se ha empleado una adaptación más sencilla de la metodología de difusión clásica, presentada por Heitz *et al.*, 2023 en su artículo: *Iterative α-(de) blending: A minimalist deterministic diffusion model*. El desarrollo del código, así como los pesos preentrenados para cada clase se pueden ver en: `https://github.com/brulonga/CloudCover.git`.

las clases intermedias lo que mejora consistentemente los resultados sobre el conjunto de test. Además, esta metodología dispone de un conjunto de test y de validación mayor, lo que dota a los resultados de una mayor fiabilidad.

Por último, cabe mencionar que para dotar a la red de aun más generalidad, se ha implementado un *data augmentation* dinámico sobre el dataset de validación y entrenamiento. Las imágenes se rotan individualmente de forma previa a entrar a la red con: 0, 90, 180 0 270 grados de forma aleatoria (cada uno con una probabilidad del 25 %) y además, se aplican *flips* verticales y horizontales con una probabilidad asignada del 40 %. De esta forma la red no ve una imagen de la misma forma cada vez que entra a la red, lo que ayuda de nuevo a combatir el *overfitting* (Capítulo 3).

Capítulo 3

Redes Neuronales

3.1. Redes Neuronales Artificiales

3.1.1. Contexto histórico

Una de las células más especiales dentro del sistema nervioso humano son las neuronas (Figura 3.1. Las neuronas están conectadas entre sí mediante axones y dendritas, y las regiones de conexión entre ambas se conocen como sinapsis (Aggarwal, 2018).

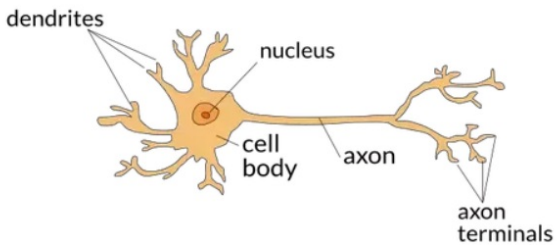

Figura 3.1: Esquema neurona biológica (Howard & Gugger, 2020)

La fuerza de las conexiones sinápticas, también conocida como "potencia sináptica" o "peso sináptico", se denomina como la efectividad con la que una neurona puede influir en otra a través del proceso sináptico. Dependiendo del impulso eléctrico dado por la neurona se libera una mayor o menor cantidad de neurotransmisores. Es decir, que pueden cambiar su nivel de activación en base al estímulo. Este fenómeno se conoce como plasticidad sináptica y es el proceso fundamental del aprendizaje neuronal, así como de la memoria ya que las neuronas se fortalecen o debilitan en base a la experiencia pasada.

En 1943, Warren McCulloch, un neurofisiólogo, y Walter Pitts, un lógico matemático, desarrollaron juntos un modelo matemático de una neurona artificial. Publicaron un artículo: *Un Cálculo Lógico de las Ideas Inmanentes en la Actividad Nerviosa* con el que asentaron las bases de las redes neuronales artificiales. En él relacionan la actividad cerebral dada por las ideas inmanentes puede con procesos lógicos (Howard & Gugger, 2020).

En su modelo lógico de la neurona, se la representa, mediante una suma simple y un sesgo o "bias" (umbral de activación), ya que para ellos la actividad de la neurona era un proceso de todo o nada. Con esto querían modelizar que una vez que el impulso eléctrico supera el potencial de acción, la neurona se activa dando lugar a una sinapsis; no hay posibilidad de "sinapsis parcial" (McCulloch & Pitts, 1958).

Sobre las bases asentadas por McCulloch y Pitts, RosenBlattd continuó el desarrollo de

la neurona artificial, dando lugar a la primera red neuronal artificial: el Perceptrón (Rosenblattd, 1962). Sin embargo, las importantes limitaciones de dicho modelo y de otros existentes, fueron demostradas poco después por Minsky y Papert (1969), lo que dio lugar a un estancamiento en el desarrollo y una paralización de la investigación durante 2 décadas.

No fue hasta 1984, en que Hopfield (Nobel 2024) propuso el algoritmo de "backpropagation" (Hopfield, 1984) que permitió superar las dificultades del perceptrón y habilitó el entrenamiento de redes mayores. Aunque en los años posteriores se continuó con la investigación, sin embargo, las limitaciones del hardware centraron la atención en otros modelos más atractivos en aquel entonces (Bosch *et al.*, 2019).

El resurgimiento de las redes neuronales hoy en día tan populares no se dio hasta 2006 en el que se publicaron nuevos algoritmos capaces de entrenar redes más profundas eficientemente. Este es el inicio de lo que se conoce como Deep Learning (Bosch *et al.*, 2019).

3.1.2. La Neurona

Una red neuronal artificial consiste en la interconexión de un conjunto de unidades elementales llamadas neuronas artificiales (Figura 3.2), llamadas a partir de aquí simplemente neuronas. En analogía a las dendritas, tenemos las conexiones de una neurona denotadas por $X = x_1, x_2, \ldots, x_n$, cuya conexión con el centro de la neurona viene evaluada por una serie de pesos $W = w_1, w_2, \ldots, w_n$. Cada peso w_j^i se puede interpretar como la importancia de la entrada x_i proveniente de la neurona j. Es decir, los pesos definen las conexiones interneuronales. Las neuronas reciben las entradas multiplicadas por sus respectivos pesos, las suman un parámetro llamado "bias" o sesgo b_j y las aplican una función de entrada o combinación, formando así en cada neurona un valor de activación z_j que será el input de la función de activación (ϕ) que define la salida de la neurona a_j. Cabe decir que la adicción del sesgo está conectada con ese valor umbral de activación de las neuronas que modelaron McCulloch & Pitts (1958) [1].

Función de entrada o combinación

Como hemos dicho, los valores de activación (z_j) se adquieren mediante la función de entrada o combinación. Se puede observar gráficamente esto en la Figura 3.2 [2]. Las función más relevante entre ellas es la suma ponderada[3] (Bosch *et al.*, 2019):

$$z(x) = \sum_{j=1}^{n} x_j w^i{}_j \tag{3.1}$$

Funciones de activación

Como hemos dicho el valor de salida de la red a_j viene dado por la aplicación de la función de activación sobre el valor de activación obtenido $f(z_j)$. La función de activación es de vital importancia para las redes neuronales, dado que, de no contar con ella, nuestras neuronas solo darían cuenta de un comportamiento lineal en el cual a medida que aumentan nuestras entradas X, aumenta nuestra salida Y; por lo que la red solo sería capaz de realizar aproximaciones lineales. Sin embargo, las funciones de activación introducen en la estructura de la neurona el comportamiento no-lineal tan fundamental para permitir a las redes neuronales modelizar

[1] Para los acrónimos o abreviaciones se ha usado el convenio utilizado por: Bosch, Anna, Jordi Casas, y Toni Lozano. *Deep Learning: Principios y Fundamentos*. Barcelona: Editorial UOC, 2019.

[2] En la Figura 3.2 se ha modificado la notación para adaptarla a la de Bosch *et al.* (2019) utilizada a lo largo de todo el trabajo.

[3] Existen otras funciones de entrada como: la función máximo, la función mínimo o las funciones lógicas AND y OR (Bosch *et al.*, 2019).

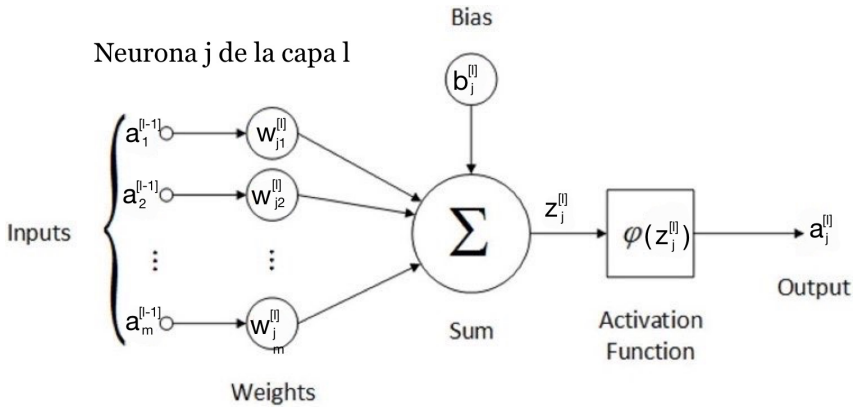

Figura 3.2: Esquema neurona artificial. En donde las entradas (inputs) vienen dados por la salida (outputs) de las neuronas de la capa $a_k^{[l-1]} \quad \forall k = 1, \ldots, m$. Cuando la entrada no proviene de otra capa, sino de los datos con los que se alimenta la red, se denomina: $a_k^{[l-1]} = x_k$. Donde x_k responden al nombre de atributos. Los pesos que se representan mediante $w_{jk}^{[l]}$ dan la relación entre la entrada k de (salida de la neurona k de la capa l-1) y la neurona j de la capa l. EL sesgo que es característico de cada neurona viene dado por $b_j^{[l]}$. Aplicando la función de entrada o combinación obtenemos nuestro valor de activación $z_j^{[l]}$ al cual si aplicamos la función de activación (que en este caso viene representada por Σ la suma ponderada) nos devuelve la salida de nuestra neurona $a_j^{[l]}$ (Farfán, 2005).

relaciones complejas en los datos. Lo que les permite aprender patrones no lineales y mejorar su capacidad de generalización (Bosch *et al.*, 2019). En el Apéndice C. Funciones de activación, se puede ver una discusión más detallada.

Arquitectura de la red

Se define a la distribución de neuronas en cada capa y de hiperparámetros [4] como la arquitectura de la red o topología de la red. Dependiendo del problema que afrontemos la arquitectura de la red óptima variará. Podemos dividir la red neuronal así pues en capas (Bosch *et al.*, 2019):

- Capa de entrada: recibe los atributos de cada instancia de entrada.
- Capas ocultas: son aquellas capas en el interior de la red. Procesan la información de la capa anterior y alimentan la entrada de la siguiente capa. Dependiendo del número de capas ocultas podemos clasificar la red como monocapa (solo una capa oculta) o multicapa (más de una capa oculta). Cuanto más profunda sea una red más capas ocultas tiene. En el caso de las redes artificiales las neuronas de una capa oculta siempre conectan con las de la siguiente capa, formando lo que se conoce como "Feed Forward Neural Nets" (FNN). Esto nos dice que la información siempre avanza en la red, por lo que no se retroalimenta (las redes neuronales recurrentes, RNN, pueden tener bucles dentro de una misma capa). Por último, si todas las neuronas de una capa están conectadas con todas las neuronas de la siguiente, entonces se las conoce como "Fully Connected Neural Nets".
- Capa de salida: esta capa es la última de la red y es la encargada de darnos nuestra predicción, ya sea una clasificación, una regresión o una segmentación. El número de

[4] Un hiperparámetro es un parámetro que se establece antes de entrenar un modelo de *machine learning* y que no se aprende directamente durante el proceso de entrenamiento. Son configurados por el usuario antes de comenzar el entrenamiento.

neuronas de nuestra capa de salida dependerá del tipo de salida que necesitemos, por ejemplo, en una clasificación binaria necesitaríamos solo una neurona, mientras que en una clasificación multiclase necesitaríamos más neuronas que nos diesen una distribución de la probabilidad de cada clase.

Podemos ver diferentes ejemplos de arquitectura de la red en la Figura 3.3:

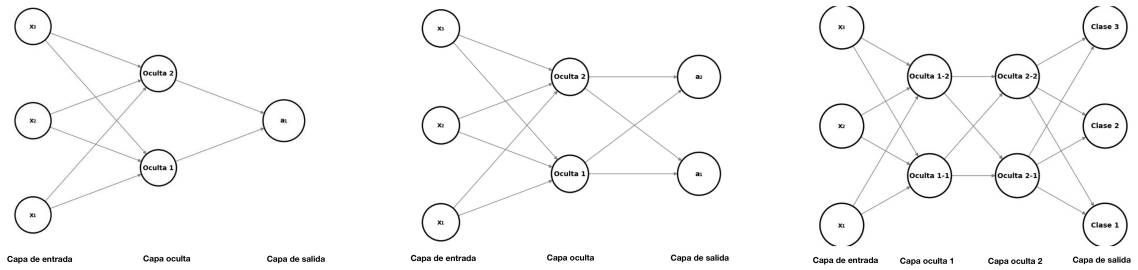

Figura 3.3: Ejemplos gráficos de diferentes arquitecturas de redes neuronales artificiales

No existe un criterio general para la elección en el número de neuronas en cada capa, sin embargo, en general la adicción de nuevas neuronas en las capas ocultas implica lo siguiente (Bosch *et al.*, 2019):

- Aumenta el poder predictivo de la red, permitiendo que la red sea capaz de reconocer patrones más complejos.
- Disminuye la posibilidad de un mínimo local (Rumelhart *et al.*, 1986).
- El tiempo de aprendizaje varía de forma inversa al número de neuronas de las capas ocultas (Plaut & Hinton, 1987).

Por el contrario, aumenta el riesgo de "overfitting" de la red, es decir, aumenta el riesgo de la sobre especialización de la red a los datos de entrenamiento, lo que reduce su valor a la hora de realizar predicciones con datos fuera del conjunto de entrenamiento.

El convenio general para dotar a la red de una primera arquitectura es dar a las capas ocultas con un número de neuronas que sea dos veces el de la capa anterior. En caso de sobreajuste reducir el número de neuronas y en caso de poca eficiencia, aumentarlo (Bosch *et al.*, 2019).

3.1.3. Entrenamiento de una red neuronal

En esta sección introduciremos el método del Descenso del Gradiente, que sienta las bases del entrenamiento de la red, reduciendo el aprendizaje de esta a un proceso de optimización de parámetros.

Función de coste

Para definir un problema de optimización debemos definir la función a optimizar. Definimos esta como la función de coste, que no difiere del error cometido por la red en la salida predicha, respecto de la salida real.

Estas son algunas de las funciones de coste más utilizadas:

- ***Error Cuadrático Medio (MSE)***:

$$y(x) = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \tag{3.2}$$

  Donde n es el número total de observaciones, y_i es el valor real y $\hat{y}_i$ es el valor predicho Comúnmente utilizada en problemas de regresión. Es sensible a errores grandes, lo que puede ser útil en ciertas aplicaciones. No se suele utilizar en tareas de clasificación ya que no captura adecuadamente la naturaleza probabilística de dicha tarea.

- ***Función entropía cruzada (Crossentropy)***:

$$H(p, q) = -\sum_{i} p(x_i)\log(q(x_i)) \tag{3.3}$$

  Donde $p(x_i)$ es la distribución de probabilidad verdadera, $q(x_i)$ es la distribución de probabilidad predicha y La suma se realiza sobre todas las clases i A diferencia del error cuadrático medio, esta función mide discrepancias entre las distribuciones de probabilidad de las predicciones y las verdaderas. Es especialmente útil para problemas de clasificación binaria y multiclase, ya que cuando la salida esta comprendida entre 0 y 1, los valores alejados de los valores esperados dan lugar a un gradiente de la función de coste muy pequeño (salida binaria función sigmoide). Con esta función se mejoran los resultados potenciando los gradientes de los extremos. Al contrario del MSE no se adapta bien a problemas de regresión con valores continuos.

- ***Categorical Cross Entropy***:

$$H(p, q) = -\sum_{i=1}^{C} y_i \log(\hat{y}_i) \tag{3.4}$$

  Donde C es el número total de clases, y_i es un valor binario que indica si la clase i es la clase correcta (1 si es la clase correcta, 0 en caso contrario) y $\hat{y}_i$ es la probabilidad predicha para la clase i. Variación de la función de Entropía Cruzada, optimizada para problemas de clasificación multiclase. Sus salidas se interpretan como distribuciones de probabilidad. Requiere que las etiquetas de clase estén en formato One-Hot[5] y su efectividad puede verse afectada en el caso de deficiencia de datos de entrenamiento de alguna de las clases.

Existen más funciones de coste como la Hinge Loss, común en Support Vector Machines (SVM), otro tipo de algoritmo de aprendizaje supervisado (Bosch *et al.*, 2019).

Método del descenso del gradiente

Una vez que tenemos definida nuestra función de coste, podemos medir como de buena es la predicción de la red neuronal para unos pesos y sesgos dados en cada neurona y reajustar los parámetros para mejorar la predicción.
Si representamos la función de coste en función de sus parámetros, obtenemos la hipersuperficie de la función de coste. En la Figura 3.4 se ha representado una función de coste arbitraria con dos parámetros para poder visualizarla en 3D. No obstante, cabe recordar que en Machine Learning trabajamos con millones de parámetros.

La primera predicción de la red, cuyos parámetros iniciales se han situado de forma

[5]One-hot: es una representación de datos categóricos en la que cada categoría se representa como un vector binario

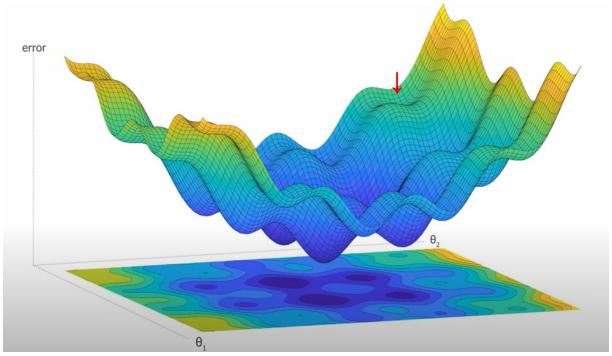

Figura 3.4: Hipersuperficie de la función de coste en 3D (DotCSV, 2018).

aleatoria, nos situará en un punto aleatorio de la hipersuperficie. Podemos evaluar la pendiente en función de la posición en la que estamos mediante las derivadas parciales en función de los parámetros. El vector que almacena la información acerca de la pendiente en cada dirección es lo que se conoce como gradiente. Este vector nos dice como tenemos que actualizar nuestros parámetros para movernos arriba o abajo sobre la hipersuperficie en una dirección dada (Dot CSV, 2018).

$$\theta = \theta - \eta \nabla f \tag{3.5}$$

Donde θ es uno de los parámetros de la red, ∇f es el gradiente de la función de coste y η es un parámetro conocido como "Learning Rate", cuya función es modular cuanto varía en cada instancia el parametro θ.
Mediante la actualización sucesiva de parámetros lo que se pretende es descender sobre la superficie hasta el mínimo global. Una de las principales adversidades que enfrentamos con los métodos de optimización basados en el descenso del gradiente es el riesgo de caer en un mínimo local. Más adelante veremos formas de minimizar este riesgo.

Backpropagation

En las redes neuronales con una sola neurona, como es el caso del perceptrón, el proceso de entrenamiento es bastante directo ya que la función de coste o el error puede expresarse explícitamente como una función de los pesos y el sesgo. No obstante, en el caso de redes multicapa, la dependencia explícita con respecto de todos los pesos de las capas anteriores se pierde por completo. Aquí es cuando requerimos de un algoritmo "backpropagation" para el calculo del gradiente, es decir, de la dependencia del error con los pesos y sesgos, de las diferentes capas y neuronas. Como dice Charu c. Aggarwal en su libro *Neural Networks and Deep Learning (2018)*:

"El algoritmo de backppropagation aprovecha la regla de la cadena del cálculo diferencial, que calcula los gradientes del error en términos de sumatorios de productos de gradientes locales a lo largo de las distintas rutas desde un nodo hasta la salida."

Se puede ver más en detalle el algoritmo de backpropagation en el Apéndice B: Backpropagation.

Desaparición del gradiente

Las redes neuronales son extremadamente sensibles al gradiente de la función de coste. Como se ha mencionado en funciones de activación en la Sección 3.1.2, una derivada nula, como la de la función ReLu para valores negativos, puede llevar a un desvanecimiento del gradiente (*vanishing gradient problem*). Esto da lugar a las llamadas neuronas muertas, que son aquellas neuronas cuya derivada de la función de activación se ha vuelto nula. Al problema opuesto,

es decir, a gradientes tan grandes que inestabilizan el proceso de entrenamiento impidiendo su convergencia, se le denomina explosión del gradiente (*exploding gradient problem*). Cuando tratamos con modelos profundos (muchas capas), estos problemas surgen, impidiendo que todas las capas se entrenen a la misma velocidad dificultando el proceso de entrenamiento. Por suerte, contamos con métodos para evitar estos problemas [6] (Bosch *et al.*, 2019).

3.2. Redes Neuronales Convolucionales

Las redes neuronales convolucionales son un tipo especial de red neuronal especializada en procesar datos con una topología similar a una cuadrícula (Goodfellow *et al.*, 2016) con dependencias espaciales fuertes en regiones locales de la cuadrícula (Aggarwal, 2018). Están inspiradas en un contexto biológico del estudio de la corteza visual de un gato (Hubel & Wiesel, 1962), en la cual partes específicas del campo visual excitaban neuronas concretas, dando los primeros indicios de una dependencia espacial localizada. Un primer modelo planteado teóricamente debido a las limitaciones de hardware de la época, fue el *Neocognitron* (Fukushima, 1980). Sin embargo, su uso no se generalizó hasta la precursora de las redes convolucionales, LeNet-5 (Le Cun *et al.*, 1989), con la cual se consiguió satisfactoriamente la clasificación de números escritos a mano. Rápidamente las redes convolucionales dominaron el marco de la visión por computador, siendo esta su principal aplicación (aunque se pueden aplicar en el análisis de datos temporales como notas de audio). En el caso de imágenes, tenemos que a las dos dimensiones espaciales, altura y grosor, se les une una tercera dimensión, la profundidad [7]. Esta dimensión da cuenta de los diferentes canales de color sobre los que se construye la imagen. En el caso de tener una imagen RGB, tendríamos una profundidad de 3, en el caso de tener una imagen en escala de grises, tendríamos una profundidad de 1 (Goodfellow *et al.*, 2016).

Se denominan redes neuronales convolucionales debido a la operación matemática que estas redes utilizan la convolución en lugar del producto matricial estándar de las redes neuronales artificiales. De hecho, se puede definir una red neuronal convolucional como (Goodfellow *et al.*, 2016):

"Las redes neuronales convolucionales son simplemente redes neuronales que utilizan convolución en lugar de multiplicación matricial general en al menos una de sus capas".

3.2.1. La operación de convolución

El núcleo de las redes neuronales son las transformaciones afines [8] a las que luego se dota de una no-linealidad mediante la función de activación. Esto es aplicable a cualquier tipo de datos de entrada, ya que por ejemplo, si tenemos una imagen 2D de 3 x 3, esta se puede descomponer a un vector 1D de 9 x 1. Sin embargo, las imágenes y otros tipos de datos tienen estructuras intrínsecas que se pierden al restringir su dimensión (Dumoulin & Visin, 2018).

Estos conjuntos de datos con propiedades especiales se almacenan en arrays multidimensionales (forma de cuadrícula). Cuentan con distintos ejes en los cuales el orden es importante y tienen una dimensión llamada profundidad (Goodfellow *et al.*, 2016) o eje de canal (Dumoulin & Visin, 2018). Estas propiedades no se desarrollan bajo las transformaciones afines, en las cuales se trata todos los ejes por igual sin tener en cuenta la información topológica. En algunos casos, como en el que nos ocupa (visión por computadora) preservar esta información topológica es

[6] Parte de este problema ya se ha abordado en la Sección 3.1.2 con las funciones de activación. Además la normalización de la distribución (*batch normalization*) (Sección 3.3.2) también ayuda a solucionar dicho problema.

[7] Aquí la palabra profundidad toma otra connotación diferente a la que tomaba en *redes neuronales profundas*, ahora no se refiere al número de capas de la red, si no a la tercera dimensión del volumen de datos.

[8] Una transformación afín entre dos espacios afines (vectoriales) consiste en una transformación lineal seguida de una traslación.

muy útil a la hora de tratar los datos.

La convolución es una operación matemática que sobre dos funciones (f y g) produce una tercera función (s) que expresa como la forma de una es modificada por la otra (Bosch *et al.*, 2019) [9].

$$s(t) = (f * g)(t) \int f(\tau)g(t-\tau)\, d\tau \tag{3.6}$$

En el contexto de las redes convolucionales la función f se refiere al *input*, la función g al filtro o *kernel* [10] que aplicamos a nuestro *input*, y la función resultante s se la denomina *mapa de características* (más adelante veremos el por qué se llama así). Tratamos conjuntos de datos discretos, por lo que definimos como convolución discreta:

$$s(t) = (f * g)(t) \sum_{n=-\infty}^{\infty} f(\tau)g(t-\tau) \tag{3.7}$$

La convolución discreta es una transformación lineal, que preserva la noción de orden en los ejes. Es localizada (opera sobre un subconjunto de los datos del input) y reutiliza los parámetros (se aplican los mismos pesos a varios subconjuntos del input) (Dumoulin & Visin, 2018).

En el caso de trabajar con imágenes, tendremos dos dimensiones de entrada, por lo que nuestro *kernel* también tendrá dos dimensiones y dado que la convolución cumple la propiedad conmutativa [11] podemos escribir:

$$s(i,j) = (I * K)(i,j) \sum_{m} \sum_{n} I(i-m, j-n)K(m,n) \tag{3.8}$$

La propiedad conmutativa se basa en el *flipping* [12] del *kernel*. No obstante, esta operación no es de relevancia para las redes neuronales [13] por lo que normalmente la mayoría de librerías implementan como convolución la función de *cross-correlation*, en la cual no se aplica este *flipping* (Goodfellow *et al.*, 2016). Se puede ver un ejemplo gráfico de convolución sin *flipping* del kernel, es decir, de *cross-correlation* en la Figura 3.5.

$$s(i,j) = (I * K)(i,j) \sum_{m} \sum_{n} I(i+m, j+n)K(m,n) \tag{3.9}$$

Las principales ventajas derivadas de la convolución son:

- ***Campo receptivo local***: en las redes neuronales convolucionales, el *kernel* con el que se aplica la convolución, es mucho más pequeño que la entrada, por lo que cada neurona de la capa *l*, solo está conectada a un subconjunto de neuronas de la *l-1* (ver Figura 3.6. A dicho subconjunto de neuronas de la capa *l-1*, se le denomina campo receptivo local (Goodfellow *et al.*, 2016).
- ***Compartición de parámetros***: a diferencia de lo visto para redes artificiales, se usan los mismos pesos y sesgos para todas las neuronas ocultas de una misma capa (Bosch *et al.*, 2019).

$$\phi(b + \sum_{l=0}^{n-1} \sum_{m=0}^{n-1} w_{l,m} a_{j+l,k+m}) \tag{3.10}$$

[9] Se denota la operación de convolución mediante el asterisco ($*$).

[10] Durante todo el trabajo se utiliza indistintamente la palabra filtro o *kernel*.

[11] $f * g = g * f$.

[12] La operación de *flipping* del *kernel* es muy similar a una transposición, se puede ver una imagen en Bosch *et al.* (2019) pag. 119.

[13] Por ejemplo, al trabajar en análisis de señales la implementación del *flipping* hace que se cumpla la propiedad asociativa, por lo que se vuelve una operación necesaria.

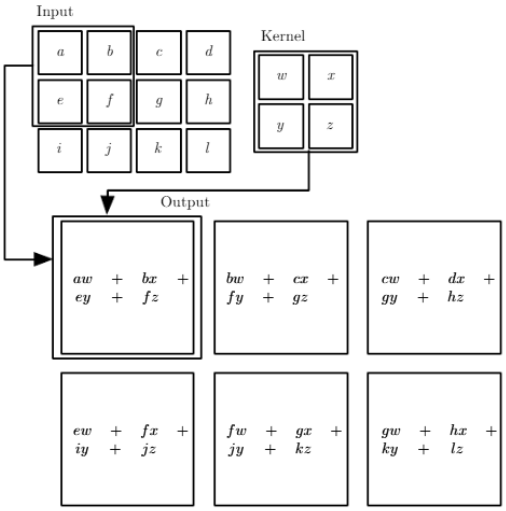

Figura 3.5: Ejemplo de una convolución 2D sin flipping del kernel (Goodfellow *et al.*, 2016)

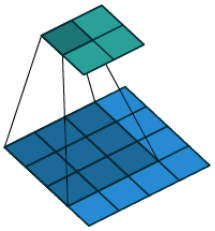

Figura 3.6: Ejemplo de campo receptivo local de 3 x 3 que se muestra en un color azul más oscuro, conectado con la neurona $(i = 1, j = 1)$ de la siguiente capa. No se debe confundir el *kernel* con el campo receptivo local. Ambos en esta imagen vienen indicados por el color azul oscuro (tienen las mismas dimensiones, tanto espaciales como de profundidad), sin embargo, son conceptos diferentes. El campo receptivo local es el subconjunto de píxeles de la imagen sobre el cual se aplica el filtro o *kernel* (Dumoulin & Visin, 2018).

Donde ϕ representa la función de activación, b es el sesgo compartido, $w_{l,m}$ es una matriz de n x n que contiene los pesos compartidos de las neuronas de la misma capa, y $a_{x,y}$ es el valor de la entrada en la posición *(x,y)*.

- ***Representaciones equivalentes***: esta propiedad se refiere al hecho de que la salida es invariante a movimientos de traslación de las entradas [14].

Lo que involucra esta compartición de parámetros aplicados sobre campos receptivos locales es que todas las neuronas detectan la misma característica en distintas zonas de la imagen. Generalmente se llama *mapa de características* a la relación de las neuronas de la capa de entrada con la capa oculta, y está conformado por los pesos compartidos y el sesgo compartido. Se dice que los pesos y sesgos compartidos nos definen un *kernel*, que tras aplicarlo a la capa de entrada nos da un *mapa de características*. Con el objetivo de detectar diferentes características, se deberán aplicar varios *kernel* sobre la entrada (Bosch *et al.*, 2019). En la Figura 3.7 podemos ver dos *mapas de características* diferentes, obtenidos mediante la aplicación de dos filtros.
En una misma capa convolucional generalmente se obtienen diferentes mapas de características. Lo cual es posible gracias a que para cada uno de ellos, los pesos y sesgos, están compartidos. Lo

[14]Un objeto desplazado unos pixeles en una imagen da lugar a una estructura muy similar de *mapas de características*, es decir, no afecta la posición del objeto dentro de la imagen.

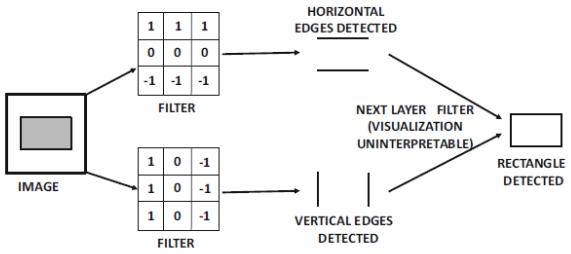

Figura 3.7: Ejemplo de dos *mapas de características* (bordes horizontales y verticales), detectados mediante los filtros en imagen, para la detección de un rectángulo (Goodfellow *et al.* 2016).

que facilita el aprendizaje de la red (hay muchos menos parámetros que aprender), permitiendo la formación de arquitecturas más profundas (ahora refiriéndonos con profundas al número de capas).

3.2.2. Capas convolucionales

Filtros o *kernel*

En las redes neuronales convolucionales, los parámetros se organizan en sets llamados filtros o *kernel*. Normalmente, el *kernel* tiene dimensiones espaciales mucho menores que la entrada sobre la que se aplica, sin embargo, *la profundidad de un filtro es siempre la misma que la de la capa sobre la que se aplica* (Aggarwall, 2018). Esta característica dimensional de los filtros se puede observar en la Figura 3.8. La operación de convolución se aplica de tal forma que el *kernel* se sitúe en cada posible posición sobre la imagen, recorriéndola entera y realizando el producto de sus $F_q \times F_q \times d_q$ parámetros sobre los campos receptivos locales (con las mismas dimensiones $F_q \times F_q \times d_q$). El *kernel* recorre la imagen de dimensiones $L_q \times B_q \times d_q$ (*Length*, *breath*, *depth*) aplicando sobre los subconjuntos de dimensión $F_q \times F_q \times d_q$ (campos receptivos locales). Si nos fijamos en la Figura 3.6, en la Figura 3.5 o en la Figura 3.8, podemos ver que la aplicación de la operación de convolución produce una reducción en la dimensión espacial tal que: $L_{q+1} = L_q - F_q + 1$ y $B_{q+1} = B_q - F_q + 1$ (Aggarwal, 2018). El número de filtros aplicados es lo que me da la profundidad de la siguiente capa (ver Figura 3.9), es decir, el número de *mapas de características* que tendré en la siguiente capa tras la convolución (Goodfellow *et al.*, 2016).

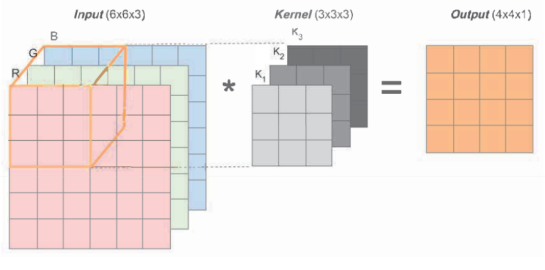

Figura 3.8: Ejemplo de aplicación de un filtro en el que se puede ver la reducción de dimensión espacial (4 = 6 -3 + 1) acorde con $L_{q+1} = L_q - F_q + 1$; y que la profundidad del filtro debe coincidir con la de la capa sobre la que se aplica (Bosch *et al.*, 2019).

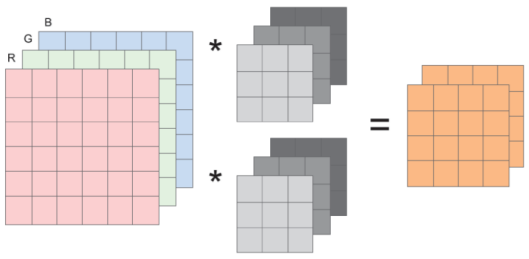

Figura 3.9: En este caso, podemos ver la aplicación de dos filtros, de donde deducimos que la profundidad de la siguiente capa será 2 (Bosch *et al.*, 2019).

Podemos modificar la expresión de la convolución discreta, para tener en cuenta la tercera dimensión, la profundidad; y los filtros o *kernel*. El p filtro de la q capa, tiene parámetros denotados por el tensor $W^{(p,q)} = w_{ijk}^{(p,q)}$ donde *i,j,k* son índices que recorren la altura, el ancho y la profundidad del filtro. Los *mapas de caracaterísticas* de la q capa se denotan por el tensor $H^{(q)} = h_{ijk}^{(q)}$. Entonces, las operaciones de convolución desde la capa q hasta la capa *q+1* son (Aggarwall, 2018):

$$h_{ijp}^{(q+1)} = \sum_{r=1}^{F_q} \sum_{s=1}^{F_q} \sum_{k=1}^{d_q} w_{rsk}^{(p,q)} h_{i+r-1,j+s-1,k}^{(q)} \tag{3.11}$$

$$\forall i \in \{1, ..., L_q - F_q + 1\} \qquad \forall j \in \{1, ..., B_q - F_q + 1\} \qquad \forall p \in \{1, ..., d_{q+1}\}$$

Realmente, las capas de convolución van seguidas de una capa ReLu que se aplica a todos los valores $L_q \times B_q \times d_q$ de la entrada de la capa (tras la convolución) creando así una cuadrícula de datos de las mismas dimensiones ($L_q \times B_q \times d_q$), pero que han pasado ya, por por el umbral de ReLu y se les añadido la no-linealidad. Estas capas normalmente no se especifican gráficamente en los modelos, sino que se incluyen dentro de las capas convolucionales.

Padding

Como vemos, la aplicación de una convolución suele reducir las dimensiones de entrada de la imagen, sin embargo, de esta forma se pierde la información en los bordes de la imagen. Para evitarlo, surge el *padding*, una aplicación mediante la cual añadimos $(F_q - 1)/2$ ceros alrededor de los bordes de la imagen para que la convolución mantenga la dimensión espacial de la imagen (Aggarwal, 2018). A este tipo de *padding* se le conoce como *half-padding*. Existe cualquier tipo arbitrario de *padding* dentro de las dimensiones del *kernel* (como se muestra en la Figura 3.10 a continuación), mediante el cual se manipula la dimensión de salida [15]. Este es el principio de acción de las convoluciones inversas [16], las cuales se realizan mediante un *full-padding* y un *kernel* del mismo tamaño que la imagen.

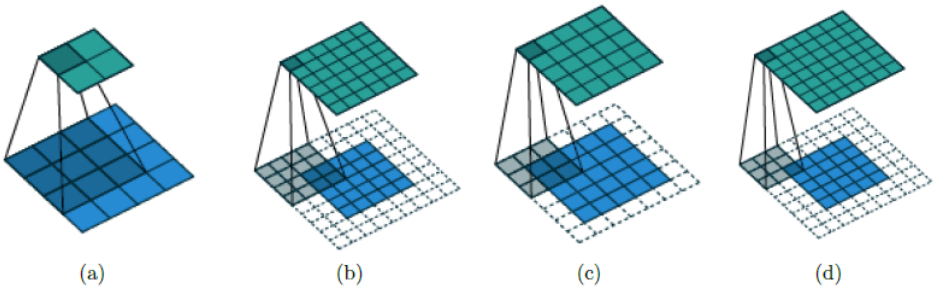

Figura 3.10: Podemos ver de izquierda a derecha: (a) aplicación de un filtro *3 x 3* con *no-padding*, (b) aplicación de un filtro *4 x 4* con *arbitrary-padding* que aumenta la dimensión de salida en este caso, (c) aplicación de un filtro *3 x 3* con *half-padding* que mantiene la dimensión de entrada, (d) *full-padding* con un filtro *3 x 3* que aumenta la dimensión de salida (Dumoulin & Visin, 2018).

[15] En la Figura 3.10 (b), se ha tomado un filtro de tamaño *4 x 4* en el caso de *arbitrary-padding*, porque un filtro *3 x 3* siempre va a caer en uno de los otros tres casos: *no-padding*, *half-padding* o *full-padding*.

[16] Las convoluciones inversas (*reverse convolution* se utilizan para obtener un tamaño de salida igual al de entrada. Generalmente se dan en redes de segmentación en las que queremos que la salida sean las capas segmentadas del mismo tamaño que la imagen (García-Pajares, 2023).

Stride

El método de *padding*, realiza convoluciones en todos los elementos de la cuadrícula. Si queremos reducir el coste computacional a expensas de perder algunas características en la salida, podemos utilizar las *convoluciones por pasos* o *strided convolutions*. Supongamos un *stride* (S_q) sobre la capa q, entonces la convolución por pasos se realizará sobre las posiciones del array multidimensional {*1,Sq+1,2Sq+1,...*} sobre las diferentes dimensiones espaciales de la imagen. Se puede ver más gráficamente en la Figura 3.11. La altura tras hacer una convolución por pasos es $(L_q - F_q)/S_q + 1$ y la anchura es $(B_q - F_q)/S_q + 1$ (Aggarwal, 2018).

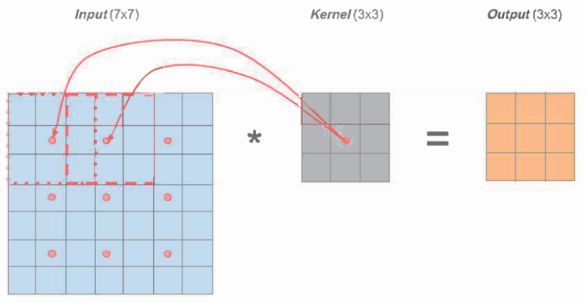

Figura 3.11: Ejemplo de una convolución por pasos sin *padding*, con un *stride* de 2. Las celdas marcadas con un punto rojo son aquellas en las que se aplicará el *kernel* (Bosch *et al.*, 2019)

Por último, podemos ver la fórmula dimensional en el caso de usar *padding* (P_q) y *stride* (S_q) al mismo tiempo (Bosch *et al.* 2019):

$$[\frac{L_q + 2P_q - F_q}{S_q} + 1] \times [\frac{B_q + 2P_q - F_q}{S_q} + 1] \tag{3.12}$$

3.2.3. Pooling

A parte de las capas convolucionales, las CNN utilizan otras capas principales dentro de su arquitectura, que son las *capas de agrupamiento* o *pooling layers*. Las capas de agrupamiento actúan sobre un pequeño subconjunto de los mapas de características ($C_q \times C_q$) y producen otra capa, con la misma profundidad (al contrario de los filtros, con lo que la profundidad del resultado depende del número de ellos aplicado).

Max-pooling (Zhou & Challappa, 1988): son las capas de agrupamiento más utilizadas. En ellas, para cada región cuadrada de ($C_q \times C_q$) de cada uno de los mapas de características (d_q) de la capa q, se toma únicamente el valor máximo. Se puede ver en la Figura 3.12 la acción de una capa *max-pooling* de *2 x 2* sobre una entrada de *4 x 4* con un *stride* de 2 (Aggarwal, 2018).

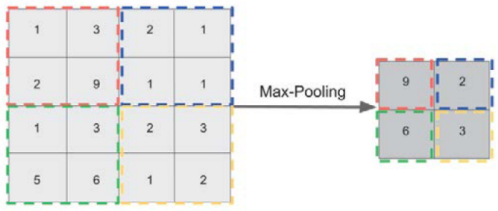

Figura 3.12: Ejemplo de la acción de una capa de *maxpooling* (Bosch *et al.*, 2019)

La salida de una capa de agrupamiento con *stride* (S_q) tendrá por dimensiones: $(L_q - C_q)/S_q + 1 \times (L_q - C_q)/S_q + 1 \times d_q$. Por lo que las capas de agrupamiento suelen disminuir drásticamente las dimensiones espaciales. A diferencia de las capas convolucionales, las capas de agrupamiento

realizan su acción sobre cada *mapa de características* para producir otro *mapa de características* (mantiene la profundidad). Mientras que la operación de convolución utiliza simultáneamente los d_q mapas de características en combinación con un filtro para producir un solo valor, y la profundidad de la siguiente capa depende solo del número filtros utilizados (Aggarwal, 2018).

3.2.4. Bloques residuales

Los bloques residuales aparecen por primera vez en la presentación de ResNet (He *et al.*, 2016). Esta red tiene una gran reputación en el marco de clasificación de imágenes. En su artículo los bloques residuales son la principal novedad arquitectónica de la red. En su artículo plantean que el *overfitting* y la desaparición del gradiente no son los principales problemas de las redes profundas. El principal problema es el gran coste temporal que supone hacer que estas redes converjan. Surgen así los bloques residuales que conectan capas a diferentes alturas mediante las llamadas *skip connections*.

$$y = F(x, W_i) + x \tag{3.13}$$

La idea tras ellos es que para la red no todos los mapas de características deben ser igual de importantes. Mediante estas *skip connections* la red puede decidir en función de los pesos que mapas deben de analizarse de forma más profunda y que mapas no requieren más que un análisis rápido (Aggarwal, 2018). Estos bloques se pueden ver de manera gráfica en la Figura 3.13.

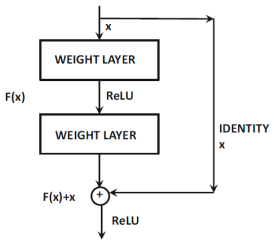

Figura 3.13: *Skip connections* en un bloque residual (Aggarwal, 2018).

De esta forma la red se optimiza para ahorrar tiempo en su entrenamiento, además de evitar la desaparición del gradiente mediante la creación de rutas alternativas para el flujo de información.

3.3. Optimización de redes neuronales

En esta sección nos centraremos en los diferentes métodos para optimizar el proceso de entrenamiento de las redes neuronales prealimentadas (*feed forward networks*) [17]. Estas redes actualizan sus parámetros mediante interacciones sobre los datos de entrenamiento, en las que aplicando el algoritmo de backpropagation optimizan la exactitud de la red (optimizan los parámetros que dan lugar a la máxima exactitud).

3.3.1. Problemas de rendimiento

Primero de todo vamos a definir unos conceptos clave:

- *Epoch* o época: se refiere a utilizar todo el conjunto de datos de entrenamiento una vez para entrenar la red neuronal. Por ejemplo, si tenemos un conjunto de 1000 datos de entrenamiento, una vez que han pasado todos una única vez cada uno, se dice que ha pasado una época.

[17] Existen problemas relacionados con la velocidad de aprendizaje, sin embargo estos no se abordan directamente en este trabajo, por lo que si se quiere incurrir en ellos se puede ver Bosch *et al.* (2019), pags. 88-91.

- *Batch*: se divide el conjunto total de datos de entrenamiento en subconjuntos de entrenamiento llamados *batch*. Los parámetros de la red no se actualizan hasta haber pasado el *batch* completo.

- Iteración: las iteraciones son el número de *batch* que hacen falta para una época, por ejemplo, si el *batch* es de 100 y tenemos 1000 datos; entonces nos harán falta 10 iteraciones.

El descenso del gradiente implementado con backpropagation es el algoritmo mediante el cual las redes neuronales optimizan sus parámetros, sin embargo, este se puede implementar de diferentes formas jugando con los conceptos recién introducidos (Bosch *et al.*, 2019).

- Descenso del gradiente estocástico (*Stochastic gradient descent*, SGD): este es el método tradicional en el cual la red actualiza los parámetros para cada ejemplo. Su principal desventaja es el alto coste computacional.

- Descenso del gradiente por lotes (*Batch gradient descent*, BGD): aunque el nombre puede llevar a la confusión, este método calcula los errores para cada ejemplo pero solo actualiza el modelo al fin de cada época. A coste de reducir mucho el coste computacional, aumenta la posibilidad de caer sobre un mínimo local.

- Descenso del gradiente por mini-lotes (*Mini-batch gradient descent*, MBGD): en este caso sí que dividimos el conjunto de entrenamiento en *batch*, y actualizamos los pesos de la red cada *batch*. Este método es el más utilizado ya que en el convergen la solidez del método estocástico y la eficiencia del BGD.

El método del descenso del gradiente es la base de la optimización de las redes neuronales, no obstante, se han desarrollado algoritmos más potentes que dan cuenta de algunos problemas de rendimiento de la red, entre ellos destacamos *Momentum* (Quian, 1999) y *Adam* (Kingma, 2014) [18].

- ***Momentum***: Este algoritmo de optimización toma la idea de que al actualizar los pesos de la red neuronal se puede tomar otro término de momento, que no solo toma en cuenta el gradiente actual, sino el gradiente acumulado de las actualizaciones pasadas. En su artículo *On the momentum term in gradient descent learning algorithms* (1999) Quian elabora la idea tras su algoritmo mediante una analogía al momento de una masa de partículas en un medio viscoso, donde las partículas tienden a seguir en movimiento en la misma dirección, lo que ayuda a suavizar las actualizaciones de los pesos.

  Al realizar su analogía, refleja el sistema de la red neuronal con un sistema de osciladores armónicos acoplados y amortiguados, que al introducir el término del momento, se acercan al amortiguamiento crítico. Es decir, la aceleración de convergencia aumenta para las dimensiones cuyos gradientes apuntan en las mismas direcciones y reduce las actualizaciones para gradientes que cambian de dirección. Se puede expresar dicho término de momento como:

$$m_t = \beta \cdot m_{t-1} + (1 - \beta) \cdot \nabla L(\theta(t)) \tag{3.14}$$

$$\theta_{t+1} = \theta_t - \eta \cdot m_t \tag{3.15}$$

[18] El estudio en profundidad de dichos algoritmos queda fuera del alcance de este trabajo, para una mayor profundidad ver los artículos originales Quian (1999), Kingma (2014), Duchi *et al.* (2011) y Zeiler (2012). También se encuentran en la sección 3.5 de *Neural Networks and Deep Learning: A Textbook* (2018) de Charu C. Aggarwall

Donde: β, es el coeficiente del *momentum* (entre 0 y 1), m_t es el momento en la actualización actual y m_{t-1} es el momento de la actualización pasada; y $\nabla L(\theta(t))$ es el gradiente de la función de coste L respecto de los parámetros θ en la actualización actual.

Como resultado, ganamos una convergencia más rápida evitando las oscilaciones en regiones con gradientes abruptos (Bosch *et al.*, 2019). Esto se puede ver de forma gráfica en la Figura 3.13.

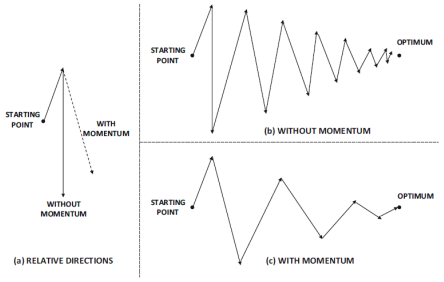

Figura 3.14: Efecto de *Momentum* al suavizar las oscilaciones del gradiente (Aggarwal, 2018, pag. 135).

- ***Algoritmos optimización adaptativos***: Los métodos de gradientes adaptativos *Adagrad* (Duchi *et al.*, 2011) y *Adadelta* (Zeiler, 2012) se basan en la variación de la tasa de aprendizaje de los parámetros dependiendo de la frecuencia de estos. Es decir, la tasa de aprendizaje se ajusta de forma inversamente proporcional a la suma acumulada de los cuadrados de los gradientes anteriores para cada parámetro. Esto significa que los parámetros que han tenido gradientes grandes en el pasado recibirán una tasa de aprendizaje más pequeña, mientras que aquellos con gradientes más pequeños recibirán una tasa mayor. *Adagrad* realiza la suma de todos los gradientes anteriores, mientras que *Adadelta* establece una ventana w de memoria de gradientes, reduciendo ostensiblemente el coste computacional.

- ***Adam, adaptative moment estimation***: *Adam* busca combinar los métodos de gradientes adaptativos con el método de *Momentum*. *Adam* funciona calculando dos momentos estadísticos a partir de los gradientes. El primer momento es la media de los gradientes, mientras que el segundo momento es la media de los cuadrados de los gradientes. Estos momentos se actualizan de forma exponencial, lo que dota de un peso mayor a los gradientes recientes. La tasa de aprendizaje varía en función de la suma de dichos momentos, lo que permite que el proceso de convergencia sea más rápido y estable. Además, incluye correcciones de sesgo para contrarrestar el efecto de iniciar los momentos en cero. Todo ello hace que *Adam* sea especialmente efectivo en problemas de optimización de redes neuronales profundas (Kingma, 2014).

3.3.2. *Overfitting*

A menudo sucede que las redes neuronales se sobre-especializan en el conjunto de entrenamiento, es decir, dan una buena exactitud sobre el conjunto de datos de prueba, pero no en el de validación. Esto ocurre porque los pesos se han sobre-ajustado a los datos de entrenamiento y la red no es capaz de generalizar, esto es lo que se conoce como *overfitting*. Existen muchas formas de evitar la sobre-especialización de la red como el *early stopping*, la regularización L2 o el tema que ya hemos tratado en el capítulo 2 de *data augmentation* o aumento de datos. No obstante, aquí nos centraremos por su común presencia dentro de las redes neuronales en la técnica de *dropout* (Scrivastava *et al.*, 2014) y *batch normalization* (Ioffe & Szegedy, 2015).

- ***Dropout***: Es una técnica de regularización[19] que involucra la desactivación de neuronas (con sus conexiones de entrada y salida) en el entrenamiento. Esto introduce ruido al proceso de aprendizaje, ayudando a evitar el *overfitting*. El proceso de Dropout se realiza asignando a cada neurona de las capas ocultas y de la capa de entrada un *dropout rate* (p_{hidden} y p_{input}). Este *dropout rate* se puede asociar a la probabilidad de cada neurona de desactivarse, en cada instancia de entrenamiento o *batch* de entrenamiento[20]. Es diferente para las neuronas de entrada que para las neuronas de las capas ocultas, siendo generalmente mayor para las neuronas ocultas. Sus valores se suelen encontrar entre 0,2 y 0,5 (Aggarwal, 2018, pag. 189). Se entrenan así un gran número de subredes con diferentes combinaciones de nodos activos, compartiendo los pesos[21] entre las redes muestreadas y entrenándose mediante conjuntos de datos muy pequeños (instancia o *batch*) (Aggarwal, 2018). El método obliga a que haya redundancia entre las características aprendidas por las unidades ocultas, lo que mejora la robustez del modelo.

- ***Batch normalization***: durante el entrenamiento las distribuciones estadísticas (media, desviación estándar) de entrada en cada capa cambian constantemente debido a la actualización de parámetros. Este fenómeno se denomina como *internal covariate shift*, y afecta a la estabilidad del proceso de entrenamiento, realentizándolo (necesidad de un *learning rate* pequeño, para que no inestabilice el proceso) y a la inicialización de parámetros. Las capas de *batch normalization* cumplen con la tarea de normalizar las distribuciones estadísticas de las entradas de la siguiente capa. Esta capa, funciona conjuntamente con el *mini-batch gradient descent* y aplica una normalización que mantiene la media cercana a cero y la desviación estándar cercana a 1 (González-Fernández, 2024). De esta forma evitamos que la red tenga que adecuarse constantemente a nuevas distribuciones, lo que acelera la velocidad de entrenamiento y nos permite usar un *learning rate* más alto así como una inicialización menos cuidadosa de pesos. Al igual, que el *dropout*, al determinar una distribución de salida de la capa, introducimos ruido al proceso de entrenamiento. Por lo que las capas de *batch normalization* actúan como un regularizador (en algunos casos eliminando la necesidad de *dropout*), además de converger en 14 pasos de entrenamiento menos (Ioffe & Szegedy, 2015).

3.3.3. Transfer Learning

El tranfer learning consiste en la reutilización de modelos previamente entrenados para la realización de una tarea parecida. Mediante la reutilización de modelos pre-entrenados se puede ahorrar un gran número de recursos computacionales. Este método es muy útil cuando las características que ha aprendido el modelo son lo suficientemente generales para su aplicación efectiva en una tarea similar (Bosch *et al.*, 2019).

Existen diferentes diferentes formas de implementar el *transfer learning* y en todas ellas hay que aplicar un *tunning* al modelo para que realice satisfactoriamente la nueva tarea específica. En general, el *tunning* del modelo variará en función de los datos de entrenamiento disponibles[22]. En nuestro caso se hará uso de la arquitectura tipo ResNet (He *et al.*, 2016), sin embargo, el entrenamiento se hará desde partiendo de cero.

[19] Las técnicas de regularización son aquellas que previenen el *overfitting*. Esencialmente son las mencionadas previamente.

[20] La técnica de *dropout* se utiliza, o bien, con métodos de entrenamiento estocásticos, o bien, de *mini-batch* (Aggarwal, 2018).

[21] Cuando se dice que se comparten los pesos durante el *dropout*, quiere decir que, si bien una combinación de neuronas esta desactivada en el *forward pass*, el backpropagation se aplica a todo el modelo, incluidas dichas neuronas que no han contribuido en el *forward pass*. El modelo entero se ajusta en cada iteracción.

[22] Se puede ver más en detalle los diferentes tipos de tunning que se pueden dar en Bosch *et al.* (2019) pag. 160-164. Siendo siempre necesario el ajuste de la capa de salida.

Capítulo 4

Entrenamiento del modelo y análisis de resultados

4.1. Arquitectura del modelo

Durante la fase de construcción del modelo se elaboraron diferentes arquitecturas de las cuales dos destacaron por sus resultados. El primer modelo al que llamaremos CNN (*Convolutional Neural Network*) consta de 5 bloques de *pooling* tal y como vemos en la Figura 4.1, enlazados con una con una capa lineal de 512 neuronas y finalmente una capa lineal de 9 neuronas que actuará como salida mediante una activación *softmax*. Este modelo es un modelo liviano en el que gracias al reducido número de convoluciones y a las capas de *max pooling*, la red tiene menos parámetros que aprender y converge de forma rápida. El modelo posee 2,5 millones de parámetros y una complejidad computacional en multiplicaciones y acumulaciones (Mac) de 87.22 MMac [1]. Como prueba de dicha medida este modelo tuvo un tiempo total de entrenamiento de 9090s empleando una única *NVIDIA RTX 3060*. Durante el entrenamiento de este modelo se observó una gran tendencia a la desaparición del gradiente, que se manifestaba en una convergencia excesivamente rápida y en una sensibilidad alta a la hora de tratar con funciones de activación como *Mish* o *Leaky ReLU*. Por ello, se desarrolló una segunda arquitectura de naturaleza residual y de mayor complejidad.

El segundo modelo, como bien se ha dicho, es un modelo residual de tipo *ResNet* (He, *et al.*, 2015). En concreto, la arquitectura principal es la de una ResNet36, donde el número 36 indica el número de bloques residuales que contiene. Cada bloque además contiene 2 convoluciones, por lo que resulta un modelo mucho más grande que el CNN. A cambio de una mayor exactitud, obtenemos un modelo más pesado, con un gran número de parámetros, 22.17 M. Su complejidad computacional supera con creces la del primer modelo, 2.68 GMac, siendo esto observado como consecuencia en un tiempo de entrenamiento total de 46368s. Al igual que el CNN la salida del modelo viene dada por dos capas *fully conected* de 512 y 9 neuronas. La salida de la última capa pasa por una activación *softmax* que normaliza la salida del modelo a una distribución de probabilidad. Se puede observar sobre Figura 4.4 un esquema de la arquitectura del modelo residual. Aunque la implementación de dicho modelo y su entrenamiento se ha realizado partiendo de cero, se considera una forma de *transfer learning*.

Debido a la sensibilidad de los modelos a la desaparición del gradiente se ha empleado como función de activación general en ambos modelos la función *Mish*. Además, se ha implementado un *dropout* sobre la capa *fully conected* de 512 neuronas con una tasa de $p = 0.4$ en ambos

[1] Se suele considerar modelos ligeros a aquellos por debajo de un GMac y modelos grandes a los que superan los 5 GMac. Para el calculo del costo computacional y del número de parámetros se ha implementado la librería de PyTorch *ptflops* (Mikulaleks, 2020)

modelos.

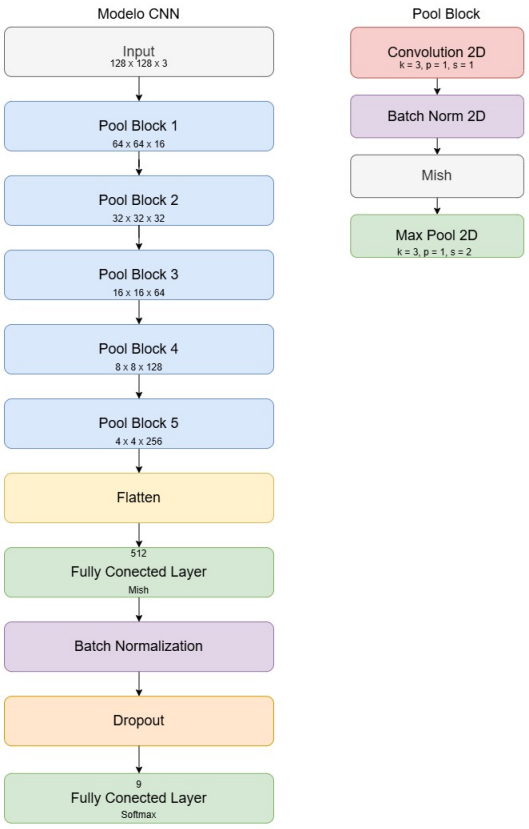

Figura 4.1: En esta figura podemos ver un esquema del modelo CNN en el cual vemos que los pilares fundamentales son los *pool blocks*, los cuales reducen de forma rápida y eficiente las dimensiones de la imagen para obtener los mapas de características. Se indican las dimensiones de los mapas de características en cada capa y sobre las *fully conected layers* podemos ver su número de neuronas y la función de activación correspondiente. Sobre las capas convolucionales k corresponde a las dimensiones del filtro (*kernel size*), p al *padding* y s al *stride*.

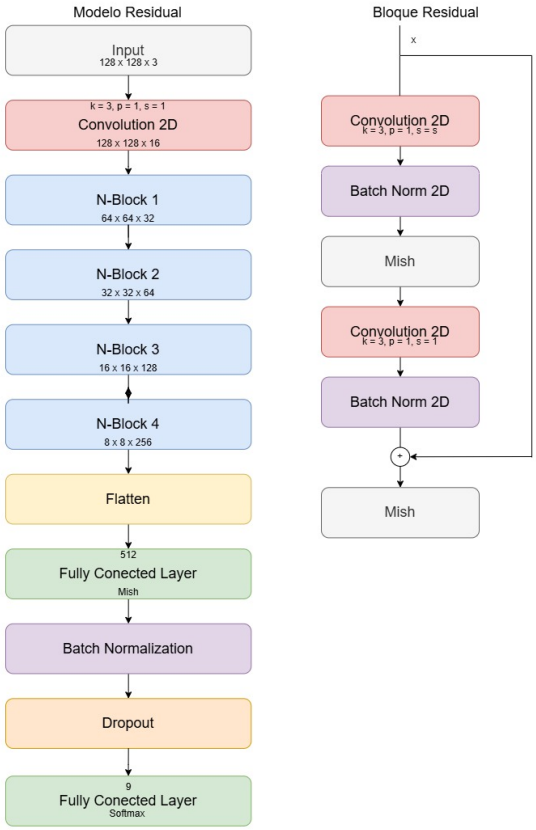

Figura 4.2: En esta segunda figura observamos el segundo modelo, el residual. Su arquitectura se basa en una ResNet (He, *et al.*, 2015). Contiene 4 *N-blocks*, los cuales son grupos de N bloques residuales como los que se observan en la figura en los cuales solo el primero reduce la dimensión espacial de los mapas de características (lo hace imponiendo $s = 2$ en la primera convolución del bloque). La suma de la entrada (x) se hace mediante una convolución 1×1 que ayuda a ajustar las dimensiones.

4.2. Entrenamiento del modelo

Para el entrenamiento del modelo se ha empleado:

- ***Función de coste***: como función de coste se ha empleado una función personalizada basada en la *Categorical Cross entropy*. En el caso de la clasificación en octas del cielo, es importante señalar a la red que una predicción errónea de una clase contigua a la clase real es más correcta que una predicción de una clase más alejada. Esto se le indica a la red mediante un peso que da cuenta de la distancia al cuadrado entre la clase predicha y la clase real.

$$H(p,q) = -\sum_{batch} |y_r - y_i|^2 y_i \log(y_i) \tag{4.1}$$

  En caso de que la clase correcta sea la predicha la perdida se hace cero y en caso de que la clase predicha sea una clase contigua a la real se recupera el caso estándar de *Categorical Cross entropy*. Esta pérdida deriva en una mejora importante en la convergencia de la red.

- ***Optimizador***: como optimizador se ha empleado el optimizador *adam* (Kingma, 2014), que combina tanto los gradientes adaptativos como el uso de *momentum*.

- ***Learning rate scheduler***: este elemento forma un papel crucial en el entrenamiento de la red y en la exactitud final que adquiera esta. En este caso se ha empleado el scheduler *One Cycle Learning Rate* (Smith, 2017, 2018, 2018). El cual, varía en cada época el *learning rate* de la forma en la que se ve en la Figura 4.3.

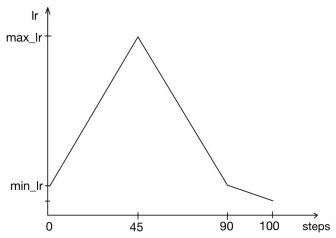

Figura 4.3: En esta imagen vemos como mediante el *scheduler One Cycle* varía el *learning rate* de forma lineal y cíclica entre un *learning rate* máximo y otro mínimo durante cada época. El *scheduler* posee una última fase en la que el *learning rate* se reduce de forma drástica.

Mediante un buen uso del *learning rate* se consigue así reducir el riesgo de *overfitting* así como reducir el tiempo de entrenamiento lo que da lugar a una exactitud mayor sobre el conjunto de validación. Es precisamente sobre el conjunto de validación sobre el que los cambios en los hiperparámetros como el *learning rate*, el número de épocas, la función de pérdida o el tipo de optimizador actúan. Es sobre este conjunto que identificamos el *overfitting* o la desaparición del gradiente.

Para el entrenamiento de la red se utilizó el lenguaje de programación Python 3.12, con la librería de diferenciación automática para inteligencia artificial PyTorch 2.5.1.[2], en el sistema operativo de linux 24.04, con CUDA 12.7 y una gpu con arquitectura Ampere como es la *NVIDIA RTX 3060*. El código completo se puede consultar en `https://github.com/brulonga/CloudCover.git`.

[2]La libreria de Pytorch actúa de forma interna mediante tensores. Por lo que, la imagen de *input* se transforma de forma predeterminada a un tensor normalizado como paso previo a la entrada a la red. De la misma manera, a la última capa, PyTorch le asigna una función de activación *softmax* por defecto.

Para el entrenamiento, como se ha dicho, se han empleado 100 épocas con un resultado final del 89.22 % de exactitud para el modelo CNN sobre el conjunto de entrenamiento y un 93.86 % sobre el conjunto de entrenamiento para el modelo *ResNet36*. Sobre el conjunto de validación se obtuvo una diferencia similar entre modelos, siendo la exactitud de validación de CNN un 63.92 % y la exactitud de validación de *ResNet36* un 72.67 %. Podemos ver la evolución de la exactitud de entrenamiento, la exactitud de validación, la perdida de entrenamiento y la pérdida de validación en función de la época en la Figura 4.4. Se puede observar claramente el cambio en la tendencia del *learning rate* en la época 45 y en la 90. El entrenamiento del modelo CNN es mucho más inestable sobre el conjunto de validación y se puede ver un poco de *overfitting* a partir de la época 90 sobre dicho conjunto. Para evitar el *overfitting* final se ha aplicado un *early stopping* que nos guarda los pesos del mejor modelo.

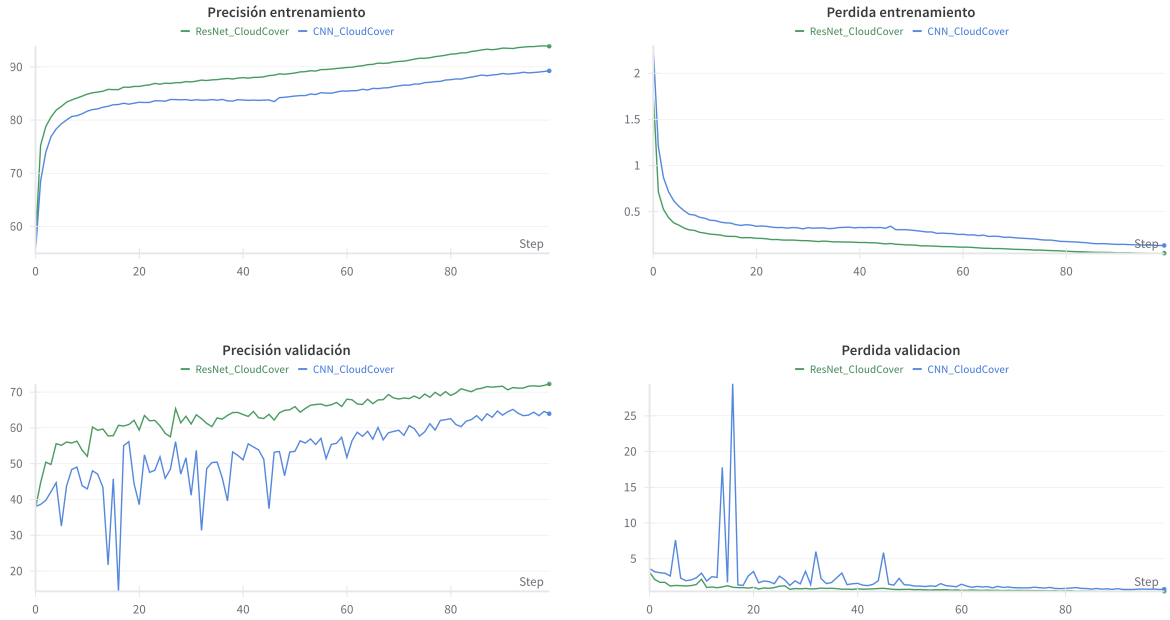

Figura 4.4: En esta figura podemos observar la evolución de la exactitud de entrenamiento, la exactitud de validación, la perdida de entrenamiento y la pérdida de validación en función de la época. Las tablas se han obtenido mediante Weights & Biases. (2024).

4.3. Análisis de resultados

Tras el entrenamiento de la red neuronal se efectúa la comprobación de su efectividad sobre el conjunto de test, conformado por 9000 imágenes reales (1000 de cada clase), que no se han usado previamente ni en el entrenamiento de la red, ni en su validación. Para ello, como primera medida de su efectividad hallamos la exactitud sobre el conjunto de test, obteniendo como resultado un 66.7 % para el modelo CNN y un 71.51 % para el modelo *ResNet36*.

Previamente a la comparación de los resultados obtenidos por otros trabajos, vamos a justificar el uso de imágenes sintéticas mediante los resultados obtenidos, además de la importancia de evaluar sobre un dataset con el mismo número de imágenes para cada clase.

Se puede observar sobre la Figura 4.5 la matriz de confusión obtenida para el modelo CNN sobre el dataset no aumentado con imágenes sintéticas con una distribución de imágenes por clase como la de la Figura 2.1.
En esta figura observamos una tasa de acierto altísima en los extremos dado que como se dijo

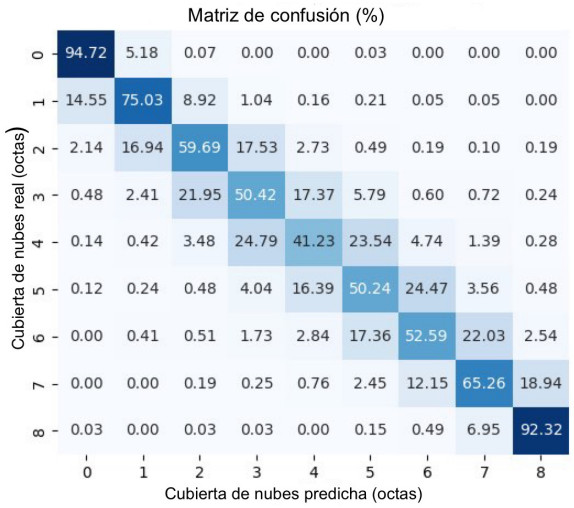

Figura 4.5: Matriz de confusión para el conjunto de test conformado por el 20 % de las imágenes del dataset sin imágenes sintéticas, del modelo CNN.

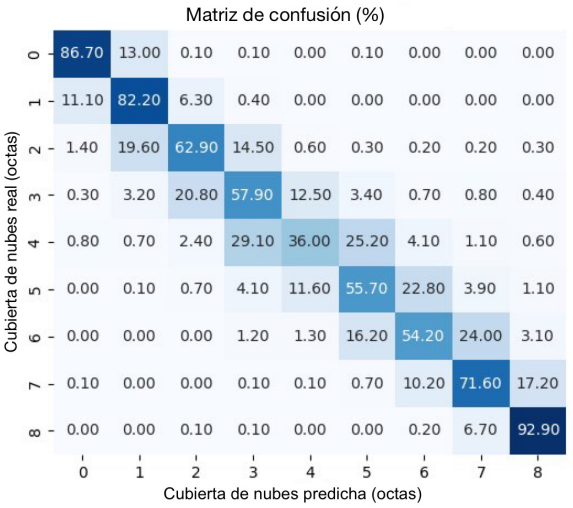

Figura 4.6: Matriz de confusión para el conjunto de test conformado por 9000 imágenes reales del dataset con imágenes sintéticas, del modelo CNN.

en el capítulo 2, las clases extremas son las más fáciles de clasificar mientras que las intermedias son las más difíciles. Dado que, el conjunto de test está conformado por un 20 % de las imágenes de cada clase, la exactitud o tasa de acierto[3] de la red viene falsificada por el desproporcionado número de imágenes de las clases extremas que son precisamente las que obtienen una mayor exactitud. De tal forma, la exactitud obtenida sobre el conjunto de test con dicha matriz de confusión fue de un 81.8 %. Esta exactitud no deja de ser ficticia, en el sentido de que si hacemos la media de la tasa de acierto por clase obtenemos un 64.61 % de exactitud y un 94.50 % de exactitud, si consideramos como acierto las clases contiguas también (tasa de acierto ± 1 octa).

Concluimos de esta forma que las tasas de acierto sobre conjuntos de test con distribuciones no constantes son más difíciles de interpretar. Además, como se ha comentado en el capítulo 2, se debe entrenar y evaluar sobre la misma distribución de ejemplos ante el riesgo de sufrir un *data shift*. Ante cualquier otra distribución de datos la exactitud vendrá sesgada y la aplicación del modelo será limitada a esa distribución en particular. Así pues, si se quiere aplicar un mismo modelo a diferentes regiones y comparar resultados, debería obtenerse el mismo número de imágenes por clase en todos los conjuntos.

Sobre la Figura 4.6, observamos el resultado mediante el uso de imágenes sintéticas. Además de estar evaluado sobre un conjunto de test mucho mayor, posee una tasa de acierto de un 66.7 % y una tasa de acierto ± 1 octas del 95.6 %. Lo que, sin duda mejora los resultados obtenidos anteriormente, a pesar de que la clase intermedia, si bien ha bajado en exactitud, la exactitud sobre el resto de clases ha aumentado considerablemente.

No obstante, una mejora consistente del 2 % sobre la exactitud en el conjunto de test puede no considerarse como suficiente justificación para el uso de imágenes sintéticas. Sin embargo, el potencial de estas imágenes sintéticas no reside en el modelo liviano CNN, si no, en que nos permiten entrenar modelos mucho más complejos que sean capaces de descifrar las características de las clases intermedias. Es aquí donde entra en juego el modelo residual.
En la Figura 4.7 podemos ver como claramente el modelo residual no consigue converger del

[3]Se define como tasa de acierto de la red: $Tasa\ de\ acierto = \frac{N\acute{u}mero\ de\ im\acute{a}genes\ clasificadas\ correctamente}{N\acute{u}mero\ total\ de\ im\acute{a}genes\ del\ conjunto\ de\ test}$

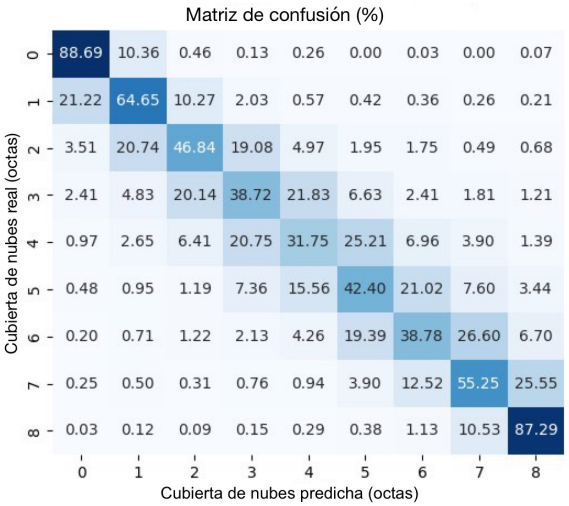

Figura 4.7: Matriz de confusión para el conjunto de test conformado por el 20 % de las imágenes del dataset sin imágenes sintéticas, del modelo *ResNet36*.

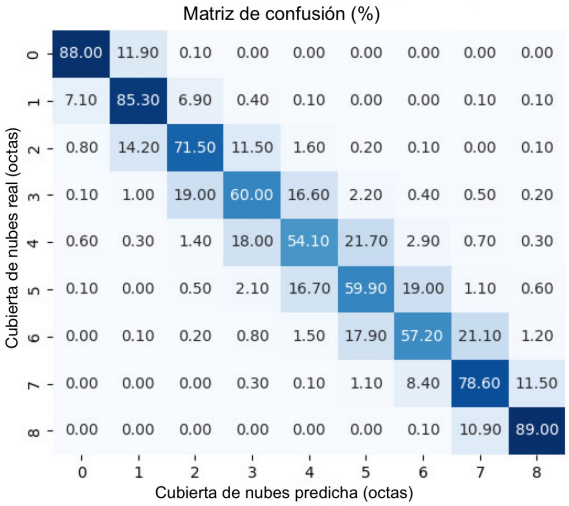

Figura 4.8: Matriz de confusión para el conjunto de test conformado por 9000 imágenes reales del dataset con imágenes sintéticas, del modelo *ResNet36*.

todo, y aunque si obtiene una tasa elevada de acierto ±1 octas (88.35 %), su tasa de acierto del 54.93 %, no es competitiva con las obtenidas para el modelo CNN.

Cuando entrenamos el modelo *ResNet36* con las imágenes sintéticas, los resultados son superiores a los obtenidos previamente. Con una tasa de acierto del 71.51 % supera por casi un 5 % al modelo CNN entrenado con imágenes sintéticas. Su mayor logro sin embargo, reside en una tasa de acierto del 54.1 % para la clase 4 octas. Esta clase es la más difícil de clasificar y de los trabajos realizados en la Universidad de Valladolid solo el de González-Fernández (2024), consigue superar el 50 % de acierto en dicha clase. El modelo residual obtiene un 97.43 % de tasa de acierto ±1 octas, superando por casi un 2 % al modelo CNN. Queda así justificada la presencia de imágenes sintéticas sobre el conjunto de entrenamiento.

Como se ha dicho previamente a la hora de comparar los resultados obtenidos con otros trabajos, se debe tener en cuenta la media de las tasas de acierto por cada clase (obtenida como la traza de la matriz de confusión dividida entre 9) y no la tasa de acierto total, que esta sesgada por la alta tasa de acierto de las clases extremas. Teniendo lo anterior en cuenta, el único trabajo comparable es el González-Fernández (2024), dado que es el único que cuenta con un dataset similar. La exactitud es altamente sensible al número de imágenes del conjunto de test, a los cirros, a los aerosoles y al criterio humano subjetivo de clasificación. En dicho trabajo se obtiene una tasa de acierto del 62.66 % y una tasa de acierto ±1 octas del 96.01 %.

Trabajos como el de García-Pajares (2024) que aborda la segmentación semántica de nubes y como consecuencia puede dar una estimación de la cubierta nubosa; o el de Calvo-Herrero (2023), que aborda datasets con un porcentaje mayor de imágenes nocturnas; no son comparables debido a la gran diferencia magnitud en el conjunto de imágenes tanto de entrenamiento, como de test. Otros trabajos con datasets menores y conjuntos de test diferentes, pero de alguna forma, más comparables, son el de Marcos-Garrachón (2024), que presenta una tasa de acierto del 58.67 % y una tasa de acierto ±1 octas del 89.44 %; y el de Alegre-Fernández (2022), que presenta una tasa de acierto del 67.3 % y una tasa de acierto ±1 del 94.83 %.

Podemos calcular parámetros estadísticos a partir de la distribución obtenida: la media (μ), el

error medio absoluto (MAE) y la desviación estándar (SD). Siendo y_i, la clase predicha; y f_i su frecuencia relativa asociada:

$$\mu = \frac{\sum_i((y_i - y_r) \cdot f_i)}{\sum f_i} \tag{4.2}$$

$$MAE = \frac{\sum_i(|y_i - y_r| \cdot f_i)}{\sum f_i} \tag{4.3}$$

$$SD = \sqrt{\frac{\sum_i[|y_i - y_r| - MAE)^2 \cdot f_i]}{\sum f_i}} \tag{4.4}$$

Obtenemos para nuestro modelo una media de $\mu = 0{,}089$ octas, un $MAE = 0{,}32$ octas y una $SD = 0{,}56$ octas. Podemos observar la distribución total de los datos en la Figura 4.9. Además podemos ver una comparativa completa de los resultados obtenidos en otros trabajos en el Tabla 4.1

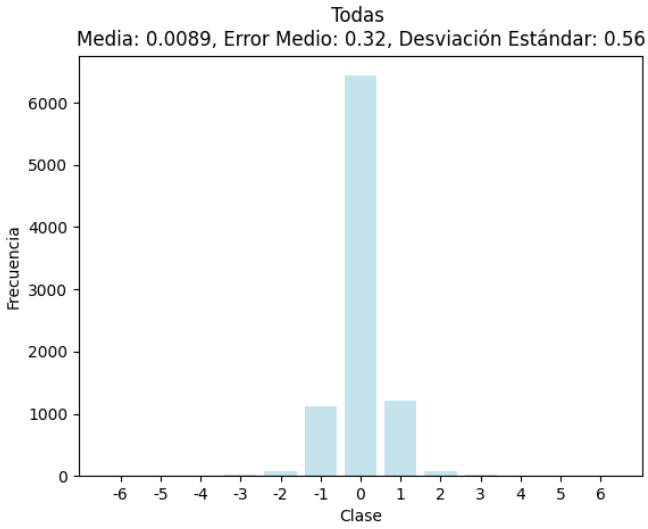

Figura 4.9: Distribución de la diferencia de octas estimadas por el modelo *ResNet36* (clase predicha) y la referencia cobertura nubosa de referencia(clase verdadera).

Tabla 4.1: Comparativa de los resultados obtenidos por los diferentes trabajos realizados en la clasificación de imágenes todo cielo para la estimación de la cubierta nubosa en octas llevados a cabo en la Universidad de Valladolid.

Modelo	**Tasa de acierto (%)**	**Tasa de acierto ±1 (%)**	**Media (μ)**	**MAE**	**SD**
CNN	66.70	95.60	-	-	-
ResNet36	**71.51**	**97.35**	**0.01**	0.32	**0.56**
González-Fernández	62.66	96.01	**0.01**	-	0.67
Marcos-Garrachón	58.67	89.44	0.03	-	0.72
Alegre-Fernández	67.30	94.83	0.05	-	0.69
García-Pajares	61.00	89.00	0.11	-	0.96
Calvo-Herrero	-	-	-0.04	-	0.84
Martinez-Celda	-	-	**-0.01**	-	1.05

Por último, podemos hacer un estudio del error medio de las predicciones de la red y de la desviación estándar predicha por clase. Se puede ver la distribución de las frecuencias en la

Figura 4.9. En la gráfica, por motivos de estética visual, se han obviado los ejemplos a mayor distancia de dos, dado que conforman un 0.66 % de los ejemplos del conjunto de test. Es decir, la red presenta un 99,34 % de tasa de acierto ±2. Los datos expuestos sobre la Figura 4.10 se pueden observar en conjunto sobre el Tabla 4.2.

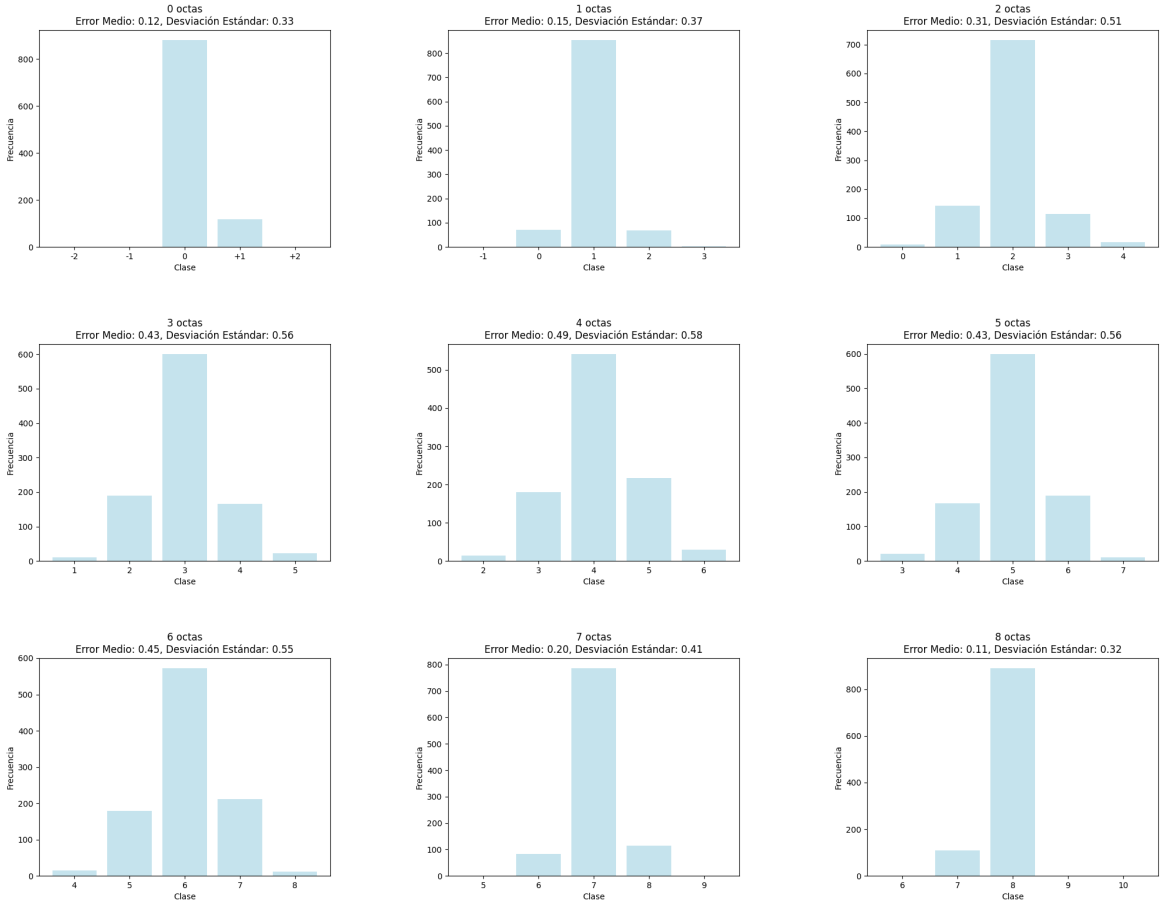

Figura 4.10: Distribución de las diferencias de la cubierta nubosa predicha por el modelo *ResNet36* y la cubierta nubosa real (calificada por un observador humano y tomada como referencia). Se puede observar además, el error medio y la desviación estándar para cada clase.

Cubierta Nubosa (oktas)	0	1	2	3	4	5	6	7	8	**Todas**
Error Medio Absoluto (MAE)	0.12	0.15	0.31	0.43	0.49	0.43	0.45	0.20	0.11	**0.32**
Desviación Estándar (SD)	0.33	0.37	0.51	0.56	0.58	0.56	0.55	0.41	0.32	**0.56**

Tabla 4.2: Error Medio Absoluto (MAE) y Desviación Estándar (SD) de la cubierta nubosa (octas) predicha por el modelo *ResNet36* y la cubierta nubosa real.

Capítulo 5

Conclusiones

En este trabajo se han empleado 71,991 imágenes obtenidas mediante siete *cámaras todo cielo* OMEA-3C y SONA202-NF, dispuestas en Valladolid, Izaña y Fuencaliente. Las imágenes empleadas en este trabajo han sido porporcionadas con su valor de cubierta de nubes (en octas) correspondiente, por el GOA-UVa[1].

Gracias al análisis de resultados obtenido de los primeros modelos se llegó a la conclusión de que, diferentes distribuciones en el conjunto de entrenamiento y el de test daban lugar a un *data shift*. Para abordar dicho problema, sin descartar un gran número de las imágenes proporcionadas por el GOA-UVa, se crearon imágenes sintéticas mediante un algoritmo de difusión determinista y de esta forma compensar las clases más desbalanceadas.

Se dividieron los datos en los conjuntos de entrenamiento, validación y test. Solo se incluirán las imágenes sintéticas sobre el conjunto de entrenamiento. que estará conformado por 99,900 imágenes (11100 por clase). Los conjuntos de validación y de test cada uno dispondrá de 9000 imágenes todas ellas reales (1000 por clase).
Se proposieron dos arquitecturas diferentes, una más liviana basada en capas de *max pooling*, y otra más compleja basada en bloques residuales. Su entrenamiento se llevó a cabo mediante una función de coste personalizada para tener en cuenta penalizaciones más elevadas en función de la distancia entre la clase predicha y la clase real; el optimizador *Adam* (Kingma, 2014) y el scheduler de *One Cycle Learning Rate* (Smith, 2017).

Los resultados obtenidos son consistentemente mejores con el conjunto de entrenamiento que incluye imágenes sintéticas. Además al disponer de un mayor número de imágenes para test (espacio muestral), aumenta la fiabilidad de las métricas. Con esto se justifica el uso de imágenes sintéticas durante en el entrenamiento.

De los resultados obtenidos se obtienen dos conclusiones en base a nuestros modelos:

- Conjuntos de entrenamiento pequeños obtienen un mejor resultados en el modelo liviano CNN.
- Conjuntos de entrenamiento que se acerquen al orden de 10^5 obtienen resultados mucho mejores mediante el uso de bloques residuales (sin cambiar la resolución espacial).

Se obtuvieron así para ambos modelos tasas de acierto satisfactorias e incluso superiores a las esperadas en el caso del modelo residual (71.51 %). El modelo *ResNet36* se destacó sobre la

[1]Sí se realizó un trabajo de clasificación de imágenes todo cielo manual, sin embargo, dichas imágenes pertenecientes a la estación meteorológica de Lidenberg, no se han podido incluir en el conjunto de imágenes del trabajo. Serán utilizadas en el futuro como ampliación del conjunto de entrenamiento aquí dispuesto.

CNN al entrenarse sobre un conjunto de imágenes mayor obteniendo los mejores resultados.

Se presentan los argumentos de porqué una distribución constante de imágenes por clase es la única distribución correcta en el proceso tanto de entrenamiento como de validación. Esta es la única forma de aplicar el modelo a cualquier clima sin sufrir *data shifts* debidos a diferencias entre distribuciones de diferentes climas.

Los puntos a mejorar en la metodología empleada son:

- Aumento en la resolución de las imágenes. Las imágenes de entrada de la red tienen una resolución de 128 x 128. Esta resolución no es suficiente para analizar estructuras tan complejas como son las nubes. Un aumento en la resolución sería un aumento en la exactitud de la detección de cirros. Estas estructuras en una resolución de 128×128 se pueden volver prácticamente indetectables al ojo humano.
- Las imágenes sintéticas nos han ayudado en este caso a mejorar nuestros resultados, sin embargo, está fuera de duda que disponer del mismo número de imágenes reales del que se ha dispuesto de sintéticas, mejoraría los resultados.
- Ampliar la captura de imágenes a otros climas que permitan a la red dar cuenta de manera más general de fenómenos climáticos como son los episodios extremos de aerosoles.
- Al emplear imágenes sintéticas el número de imágenes nocturnas se reduce ampliamente. En la creación de imágenes si que se obtienen imágenes con diferentes exposiciones sin embargo debido al bajo número de imágenes nocturnas el proceso de difusión no logra abarcar este tipo de imágenes. Es decir las imágenes sintéticas son diurnas.
- Evaluar un un modelo entrenado con imágenes en una región (siempre con las mismas imágenes por clase para evitar *data shifts*) en regiones con climas radicalmente diferentes y analizar los resultados; o crear un modelo generalizado con imágenes de otros climas que de cuenta de todo tipo de fenómenos.
- De una forma más técnica se podría probar a añadir en la arquitectura de la red bloques de atención (Vaswani, 2023) en los cuales se basan las redes *transfromers* (como ChatGPT). Esto, junto con un conjunto de datos aun más amplio permitiría a la red obtener información posicional así como características más abstractas cuanto más profunda sea.

Ambos modelos, así como el empleado en la difusión, se encuentran de forma gratuita en el repositorio de GitHub: `https://github.com/brulonga/CloudCover.git`. Los pesos para la creación de imágenes sintéticas y para la clasificación de imágenes todo cielo se encuentran en dicho repositorio también. Se anima al lector a comprobar los resultados obtenidos por su cuenta.

Bibliografía

Aggarwal, Charu C. *Neural Networks and Deep Learning: A Textbook*. Nueva York: Springer, 2018.

Ahrens, C. Donald. *Meteorology today: an introduction to weather, climate, and the environment*. Cengage Learning Canada Inc, 2015.

Alegre Fernández, Sergio. *Clasificación automática de imágenes del cielo mediante inteligencia artificial* . Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2022.

Alonso-Montesinos, J., Batlles, F.J., Portillo, C.. *Solar irradiance forecasting at one-minute intervals for different sky conditions using sky camera images*, Energy Conversion and Management, Volume 105, 2015, Pages 1166-1177, ISSN 0196-8904.
https://doi.org/10.1016/j.enconman.2015.09.001

Arking, A., Childs, J.D., Merritt, J.. Remote Sensing of Cloud Cover Parameters. In: *Liou, KN., Xiuji, Z. (eds) Atmospheric Radiation*. American Meteorological Society, Boston, MA, 1987.
https://doi.org/10.1007/978-1-935704-18-8_70

Antuña, J. C.. *Configuración y metodología para el uso de cámaras todo cielo en la obtención de parámetros atmosféricos*. Valladolid, Universidad de Valladolid, Tesis Doctoral, 2021.

Antuña-Sanchez, J. C., Diaz, N., Estevan, R., de Frutos, A., and Antuña-Marrero, J. C. (2015). Cloud camera design using a Raspberry Pi. Optica Pura y Aplicada, 48(3), 199– 205.

Barbieri, F., Rajakaruna, S., Ghosh, A., 2017. *Very short-term photovoltaic power forecasting with cloud modeling: A review*. Renew. Sustain. Energy Rev. 75, 242–263.

Bayer, B. E. (1976). *Color imaging array*.

Bosch, Anna; Casas, Jordi; y Toni Lozano. *Deep Learning: Principios y Fundamentos*. Barcelona: Editorial UOC, 2019.

Boucher, O.; Randall, D.; Artaxo, P.; Bretherton, C.; Feingold, G.; Forster, P.; Clouds and Aerosols. In: *Climate Change 2013 – The Physical Science Basis: Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge, Cambridge University Press, págs. 571-657.

Boyle,W. S. and Smith, G. E. (1970). *Charge Coupled Semiconductor Devices*. Bell

System Technical Journal, 49(4), 587–593.

Calvo Herrero, Carolina. *Clasificación automática de imagenes del cielo mediante Inteligencia Artificial*. Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2023.

Campbell, James B., and Randolph H. Wynne. *Introduction to remote sensing*. Guilford press, 2011.

Casanova Roque, Carlos. *Apuntes de termodinámica*. Valladolid, Universidad de Valladolid, 2018.

Cazorla, A., Olmo, F.J., Alados-Arboledas, L.. *Using a Sky Imager for aerosol characterization, Atmospheric Environment, Volume 42, Issue 11, 2008, Pages 2739-2745.* https://doi.org/10.1016/j.atmosenv.2007.06.016

Chao Li, Jinji Ma, Peng Yang, Zhengqiang Li, *Detection of cloud cover using dynamic thresholds and radiative transfer models from the polarization satellite image*, Journal of Quantitative Spectroscopy and Radiative Transfer, Volumes 222–223, 2019, Pages 196-214, ISSN 0022-4073.
https://doi.org/10.1016/j.jqsrt.2018.10.026https://doi.org/10.1016/j.jqsrt.2018.10.026.

Clothiaux, E. E., M. A. Miller, B. A. Albrecht, T. P. Ackerman, J. Verlinde, D. M. Babb, R. M. Peters, and W. J. Syrett. *An evaluation of a 94-GHz radar for remote sensing of cloud properties. Journal of Atmospheric and Oceanic Technology 12, no. 2 (1995): 201-229.*

Debevec, P. E. and Malik, J. (1997). *Recovering high dynamic range radiance maps from photographs*. In Proceedings of the 24th annual conference on Computer graphics and interactive techniques - SIGGRAPH '97, pages 369–378, Not Known. ACM Press.

DotCSV. (2018). *¿Qué es el Descenso del Gradiente? Algoritmo de Inteligencia Artificial — DotCSV*. YouTube, Febrero, 4.
https://www.youtube.com/watch?v=A6FiCDoz8_4

Duchi, John, Hazan, Elad and Singer, Yoram. 2011. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. J. Mach. Learn. Res. 12, null (2/1/2011), 2121–2159.

Dumoulin, Vincent y VISIN, Francesco, 2016. *A guide to convolution arithmetic for deep learning.*
https://arxiv.org/abs/1603.07285

Farfán, Fernando. (2005). *Control Cerebral de Interfases: Análisis Exploratorio de Técnicas Paramétricas Digitales para la Detección y Cuantificación de Estados Mentales.*

Forster, P.; Storelvmo, T.; Armour, K.; Collins, W.; Dufresne, J. L.; Frame, D.; Lunt, D. J.; Mauritsen, T.; Palmer, M. D.; Watanabe, M.; Wild, M. y Zhang, H.. *The Earth's Energy Budget, Climate Feedbacks and Climate Sensitivity.* Cambridge, Cambridge

University Press, 2023.

Frisch-Niggemeyer, Anita, Philipp Weihs, Michael Revesz, Stefan F. Schreier, Andreas Richter, *Relating atmospheric aerosol amounts to blue to red ratio and grayscale contrast fluctuations using digitalization of routine webcam photographs taken in the urban environment of Vienna*, Atmospheric Environment, Volume 290, 2022, 119345, ISSN 1352-2310.
https://doi.org/10.1016/j.atmosenv.2022.119345

Fukushima, K. *Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position.* Biol. Cybernetics 36, 193–202 (1980).
https://doi.org/10.1007/BF00344251

García Pajares, Sergio. *Segmentación semántica multicategoría de imágenes todo cielo mediante redes neuronales.* Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2024.

Ghonima, M. S., Urquhart, B., Chow, C. W., Shields, J. E., Cazorla, A., and Kleissl, J.: *A method for cloud detection and opacity classification based on ground based sky imagery*, Atmos. Meas. Tech., 5, 2881–2892, 2012.
https://doi.org/10.5194/amt-5-2881-2012

Gutiérrez, J.M., Cano, R., Cofiño, A.S., Sordo, C., 2004. *Redes Probabilísticas y Neuronales en las Ciencias Atmosféricas (Probabilistic and Neural Netwoks on Atmospheric Sciences).* Ministerio de Medio Ambiente. Dirección General del Instituto Nacional de Meteorología.

González-Fernández, D.; Román, R.; Antuña-Sanchez, J.C.; Cachorro, V.E.; Copes, G.; Herrero-Anta, S. *et al.. A neural network to retrieve cloud cover from all-sky cameras: A case os study over Antartica.* Quarterly Journal of the Royal Meteorological Society, 1-19, 2024.
https://doi.org/10.1002/qj.4834

Goodfellow, Ian and Bengio, Yoshua and Courville, Aaron. *Deep Learning.* MIT Press, 2016.
http://www.deeplearningbook.org

He, Kaiming, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "*Deep residual learning for image recognition.*Ïn Proceedings of the IEEE conference on computer vision and pattern recognition, pp. 770-778. 2016.
https://arxiv.org/abs/1512.03385

Heitz, Eric, Laurent Belcour, and Thomas Chambon. *Iterative α-(de) blending: A minimalist deterministic diffusion model.* In ACM SIGGRAPH 2023 Conference Proceedings, pp. 1-8. 2023..
https://arxiv.org/abs/2305.03486

Hopfield, J. J. (1984). Neurons with Graded response Have Collective Computational Properties Like Those os Two-State Neurons. Proceedings of the National Academy of Sciences (vol. 81, nº10, pags. 3088-3092).

Howard, Jeremy, y Sylvain Gugger. *Deep Learning for Coders with FastAI & PyTorch*. Sebastopol: O'Reilly Media, 2020.

Howard, Luke . *The Climate of London* . London: G. and W. Nicol, 1803.

Hubel, David H. and Torsten N. Wiesel. "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex." The Journal of Physiology 160 (1962): n. pag.
https://api.semanticscholar.org/CorpusID:17055992

Huo, J. & Lu, D. (2012) *Comparison of cloud cover from all-sky imager and meteorological observer*. Journal of Atmospheric and Oceanic Technology, 29, 1093–1101.
https://doi.org/10.1175/JTECH-D-11-00006.1

Ioffe, S. & Szegedy, C. (2015) *Batch normalization: accelerating deep network training by reducing internal covariate shift*. arXiv.
https://arxiv.org/abs/1502.03167v3

Jacobson, Mark Z. *Fundamentals of Atmospheric Modeling*. Nueva York, Cambridge University Press, 2005.

Kim, B.-Y., Jee, J.-B., Zo, I.-S. & Lee, K.-T. (2016) *Cloud cover retrieved from skyviewer: a validation with human observations*. Asia-Pacific Journal of Atmospheric Sciences, 52, 1–10.

Kingma, Diederik P. .ªdam: A method for stochastic optimization..ªrXiv preprint arXiv:1412.6980 (2014). https://doi.org/10.48550/arXiv.1412.6980

Kreuter, A., Zangerl, M., Schwarzmann, M., and Blumthaler, M.. *All-sky imaging: a simple, versatile system for atmospheric research*, Appl. Opt. 48, 1091-1097 (2009).

Krizhevsky, A., Sutskever, I. & Hinton, G.E. (2012) *ImageNet classification with deep convolutional neural networks*. En: Pereira, F., Burges, C.J.C., Bottou, L. & Weinberger, K.Q. (Eds.) Proceedings of the 25th International Conference on Neural Information Processing Systems. Red Hook, NY. Available from:: Curran Associates, Inc, pp. 1097–1105.

Kshudiram, Saha. *The Earth's Atmosphere. Its Physics and Dynamics*. Berlín, Springer, 2008.

Le Cun, Boser, Denker, Henderson, Howard, Hubbard and Jackel. (1989). *Handwritten Digit Recognition with a Back-Propagation Network*. Neural Computation (vol. 1, nº 2, pags. 1929-1958).

Marcos Garrachón, Víctor. *Clasificación automática de imagenes del cielo mediante Inteligencia Artificial*. Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2024.

Martin Šinko, Peter Sýkora, Patrik Kamencay, Róbert Hudec, Development of a

system for collecting and processing sky images and meteorological data used for weather prediction, Transportation Research Procedia, Volume 40, 2019, Pages 1548-1554, ISSN 2352-1465.
https://doi.org/10.1016/j.trpro.2019.07.214

Martínez Celda, Bernardo. *Clasificación automática de Imágenes del Cielo mediante Inteligencia Artificial*. Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2021.

Martinez, Isidoro. *Termodinámica de la Atmósfera*. Online, Universidad Politécnica de Madrid, 1995.

Masuda, Ryosuke, Hironobu Iwabuchi, Konrad Sebastian Schmidt, Alessandro Damiani, and Rei Kudo. 2019. *Retrieval of Cloud Optical Thickness from Sky-View Camera Images using a Deep Convolutional Neural Network based on Three-Dimensional Radiative Transfer* Remote Sensing 11, no. 17: 1962.
https://doi.org/10.3390/rs11171962

Matos, M., Bessa, R., Botterud, A., Zhou, Z., 2017. *Forecasting and setting power system operating reserves, Renewable Energy Forecasting: From Models to Applications*

McCulloch, Warren S., y Walter Pitts. *Un Cálculo Lógico de las Ideas Inmanentes en la Actividad Nerviosa*. Chicago: Editorial Nueva Visión, 1958.

Mikulaleks. (2020). *ptflops: Flops counter for PyTorch models*. GitHub.
https://github.com/sovrasov/flops-counter.pytorch

Minsky, M.; Papert, S. (1969). *Perceptrons: An Introduction to Computational Geometry*. Cambridge: MIT Press.

Misra, Diganta. "Mish: A self regularized non-monotonic activation function..ªrXiv preprint arXiv:1908.08681 (2019).
https://arxiv.org/abs/1908.08681

Muhammad Syazwan Rifdi Bin Mohd Rashid, Jinghong Zheng, Ernest Sng, Kurinji Malar Rajendhiran, Zhen Ye, Li Hong Idris Lim, *An enhanced cloud segmentation algorithm for accurate irradiance forecasting*, Solar Energy, Volume 221, 2021, Pages 218-231, ISSN 0038-092X.
https://doi.org/10.1016/j.solener.2021.03.061

OMM, O.M.M.. *International Cloud Atlas: Manual on the observation of clouds and other meteors*. Online, 2017.
https://cloudatlas.wmo.int/es/home.html

Plaut, David C., Steven J. Nowlan and Geoffrey E. Hinton. *Experiments on Learning by Back Propagation*. (1986).
https://api.semanticscholar.org/CorpusID:15150815

Qian, Ning. *On the momentum term in gradient descent learning algorithms*, Neural Networks, Volume 12, Issue 1, 1999, Pages 145-151, ISSN 0893-6080.

https://doi.org/10.1016/S0893-6080(98)00116-6

Quirantes, Jose A. y Jose A. Gallego. *Atlas de Nubes y Meteoros*. Torrelavega, Editorial de Urueña, Castilla Tradicional S.L. y Cantabria tradicional S.L., 2011.

Ricciardelli, Elisabetta & Romano, Filomena & Cuomo, V.. Physical and statistical approaches for cloud identification using Meteosat Second Generation-Spinning Enhanced Visible and Infrared Imager Data. En: *Remote Sensing of Environment - REMOTE SENS ENVIRON. 112. 2741-2760.* 2008.
https://doi.org/10.1002/qj.4834

Román, R., Cazorla, A., Toledano, C., Olmo, F.J., Cachorro, V.E., de Frutos, A., Alados-Arboledas L.. *Cloud cover detection combining high dynamic range sky images and ceilometer measurements*, Atmospheric Research, Volume 196, 2017, Pages 224-236, ISSN 0169-8095.
https://doi.org/10.1016/j.atmosres.2017.06.006

Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-net: Convolutional networks for biomedical image segmentation.* In Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18 (pp. 234-241). Springer International Publishing.
https://arxiv.org/abs/1505.04597

Rosenbladdt, F. (1962). *Principles of Neurodynamics; Perceptrons and the Theory of Brain Mechanisms*. Spartan Books.

Rossow, William B.; Shiffer, Robert A.. Advances in Understanding Clouds from ISCCP. In: *Bulletin of the American Meteorological Society (BAMS), Volume 80, Issue 11, pag. 2261 - 2288.* American Meteorological Society, Boston, MA, 1999.
https://doi.org/10.1175/1520-0477(1999)080<2261:AIUCFI>2.0.CO;2

Rumelhart, David E., Geoffrey E. Hinton and Ronald J. Williams. *Learning representations by back-propagating errors*. Nature 323 (1986): 533-536.

Ryo Onishi, Daisuke Sugiyama, Deep Convolutional Neural Network for Cloud Coverage Estimation from Snapshot Camera Images, SOLA, 2017, Volume 13, Pages 235-239, Released on J-STAGE December 21, 2017, Online ISSN 1349-6476.
https://doi.org/10.2151/sola.2017-043

Sanz Huidobro, Sergio. *Análisis de imágenes de cielo con una inteligencia artificial* . Valladolid, Universidad de Valladolid, Trabajo de Fin de Grado, 2023.

Shi, M., Xie, F., Zi, Y. and Yin, J.. *Cloud detection of remote sensing images by deep learning*, 2016 IEEE International Geoscience and Remote Sensing Symposium (IGARSS), Beijing, China, 2016, pp. 701-704, doi: 10.1109/IGARSS.2016.7729176.

Sidenbladh, Hedvig; Black, Michael J.; and David J. Fleet. *Seguimiento Estocástico de Figuras Humanas 3D Usando Movimiento de Imágenes 2D*. Stockholm & Coyote Hill, Springer, 2000.

Smith, Leslie N. "*Cyclical learning rates for training neural networks.*Ïn 2017 IEEE winter conference on applications of computer vision (WACV), pp. 464-472. IEEE, 2017.
https://arxiv.org/abs/1506.01186

v Smith, Leslie N., and Nicholay Topin. "Super-Convergence: Very Fast Training of Neural Networks Using Large Learning Rates." ArXiv:1708.07120 [Cs, Stat], May 17, 2018.
http://arxiv.org/abs/1708.07120

Smith, Leslie N. "A Disciplined Approach to Neural Network Hyper-Parameters: Part 1 – Learning Rate, Batch Size, Momentum, and Weight Decay." ArXiv:1803.09820 [Cs, Stat], April 24, 2018.
http://arxiv.org/abs/1803.09820

Srivastava, Nitish; Hinton, Geoffrey; Krizhevsky, Alex; Sutskever, Ilya; and Salakhutdinov, Ruslan. 2014. *Dropout: a simple way to prevent neural networks from overfitting.* J. Mach. Learn. Res. 15, 1 (January 2014), 1929–1958.

Tapakis, Rogiros and Alexandros G. Charalambides. *Equipment and methodologies for cloud detection and classification: A review.* In Solar Energy 95 (2013): 392-430. 2013.
https://doi.org/10.1016/j.solener.2012.11.015

Valliappa Lakshmanan, Martin Görner, Ryan Gillard. *Aprendizaje automático práctico para visión por ordenador.* O'Reilly Media, 2024.

Vaswani, A. .Attention is all you need..Advances in Neural Information Processing Systems (2017).
https://arxiv.org/abs/1706.03762

Visconti, Guido. *Fundamentals of Physics and Chemistry of the Atmosphere.* Suiza, Springer, 2016.

Wallace, John M. y Peter V. Hobbs. *Atmospheric Science: An Introduction Survey.* San Diego, Elvesier, 2006.

Weights & Biases. (2024). Weights & Biases
https://wandb.ai

Weng, Qihao, ed. *Advances in environmental remote sensing: sensors, algorithms, and applications.* CRC Press, 2011.

Xie, W., Liu, D., Yang, M., Chen, S., Wang, B., Wang, Z. et al. (2020). *SegCloud: a novel cloud image segmentation model using a deep convolutional neural network for ground-based all-sky-view camera observation.* Atmospheric Measurement Techniques, 13, 1953–1961.
https://doi.org/10.5194/amt-13-1953-2020

Xu, Bing. .Empirical evaluation of rectified activations in convolutional network..arXiv preprint arXiv:1505.00853 (2015).
https://arxiv.org/abs/1505.00853

Yau, M.K., and R.R. Rogers. 1989. *A Short Course in Cloud Physics*. 3rd ed. San Diego: Academic Press.

Yinsen Niu, Jifeng Song, Lianglin Zou, Zixuan Yan, Xilong Lin, *Cloud detection method using ground-based sky images based on clear sky library and superpixel local threshold*, Renewable Energy, Volume 226, 2024, 120452, ISSN 0960-1481.
https://doi.org/10.1016/j.renene.2024.120452

Zeiler, Matthew D. .ADADELTA: an adaptive learning rate method..arXiv preprint arXiv:1212.5701 (2012). https://doi.org/10.48550/arXiv.1212.5701

Zhao, G., and L. Di Girolamo. *Cloud fraction errors for trade wind cumuli from EOS-Terra instruments*, Geophys. Res. Lett., 33, L20802. 2006.
https://doi.org/10.1029/2006GL027088

Zhao Zhen, Zheng Wang, Fei Wang, Zengqiang Mi, Kangping Li, *Research on a cloud image forecasting approach for solar power forecasting*, Energy Procedia, Volume 142, 2017, Pages 362-368, ISSN 1876-6102,
https://doi.org/10.1016/j.egypro.2017.12.057

Zhou, Yi-Tong and Rama Chellappa. "Computation of optical flow using a neural network." IEEE 1988 International Conference on Neural Networks (1988): 71-78 vol.2.

Apéndice A

Cuestiones acerca de la formación de las nubes

A.1. La Atmósfera terrestre

Según la Organización Meteorológica Mundial (OMM) se define la atmósfera como:

"La atmósfera es una capa de gases que rodea un planeta o un satélite, mantenida por la gravedad, que interactúa con la superficie del planeta y que es esencial para los procesos meteorológicos y climáticos."

Debido a la capacidad para comprimirse de la atmósfera terrestre, se encuentra muy estratificada verticalmente y, aunque dependiendo del fenómeno a estudiar podemos delimitar diversas capas de interés, suelen considerarse cuatro capas en función del perfil vertical de temperatura (Martínez, 1995):

Troposfera La troposfera es la capa más baja de la atmósfera y se extiende desde la superficie terrestre hasta una altura entre 8 y 15 km. Aunque parezca una medida burda, no podemos dar un único valor a su grosor, ya que depende de la zona geográfica. Es más gruesa en el ecuador y más delgada en los polos. En general la superficie de separación con la siguiente capa, llamada la tropopausa, varía mucho. Varía para latitudes entre 30^{o} y 60^{o} (llamándola discontinua en esas latitudes), así como varía con la temperatura, es decir con la estación. Es la capa más extensa conteniendo tres cuartas partes de la masa atmosférica total y un 99 % del vapor de agua total. Se compone por nitrógeno (78 %), oxígeno (21 %), gases traza (dioxido de carbono, metano, óxidos de nitrógeno, clorofluorocarbonos, compuestos orgánicos volátiles y dióxido de azufre), vapor de agua (cuyo porcentaje varía entre el 1 y el 4), aerosoles y de las nubes. Una de las principales características de la troposfera es el gradiente negativo de la temperatura con la altura, siendo máxima en la superficie terrestre y disminuyendo $6{,}5°C$ por kilómetro, llegando así a los $-70°C$. Esta disminución de la temperatura en el perfil vertical (ver Figura A.1) sienta las bases para la formación de nubes y otros fenómenos meteorológicos. De hecho, es en la troposfera donde ocurren la mayor parte de fenómenos meteorológicos, convirtiéndola así en la capa más dinámica que alberga los sistemas de alta y baja presión.

Estratosfera La división térmica hecha indica un cambio en el gradiente negativo de la temperatura. Tras la tropopausa, la temperatura se estabiliza y se mantiene constante, para volver a ascender progresivamente y alcanzar una temperatura de hasta $0°C$ en la estratosfera, que alberga una altura desde la tropopausa hasta los 50km, altura marcada por la estratopausa, región que delimita el paso a la mesosfera. Su característica más importante es el balance radiativo, y es que, en esta capa entre los 15km y los 30 km, se encuentra la capa de ozono u ozonosfera. Esta capa es la encargada de absorber la radiación en el espectro ultravioleta proveniente del sol

que es tan perjudicial para la vida en la tierra (Marcos-Garrachón, 2024), actuando así, como una pantalla protectora. El perfil vertical estable de temperaturas de la estratosfera previene las corrientes de aire verticales, así como una presión atmosférica mucho menos intensa. Esto permite precisamente la formación de la capa de ozono en un ambiente mucho más estable, climáticamente hablando, que la troposfera, habiendo corrientes moderadas y horizontales.

Mesosfera Su importancia reside en que es el escudo protector terrestre contra meteoroides, desintegrándose la mayoría en esta capa. Se extiende desde la estratopausa, hasta la mesopausa situada a unos 80km de altura. En ella de nuevo vemos el gradiente negativo de la temperatura con la altura. En ella comienzan a aparecer los primeros iones por descomposición solar de los óxidos de nitrógeno.

Termosfera o ionosfera Abarca desde la mesopausa hasta los 500km de altura. El perfil vertical de la temperatura vuelve a cambiar, manteniéndose primero constante en -80^{o}C o -90^{o}C, para crecer asintóticamente hasta más de 1000K. Esto se debe a la absorción de la radiación solar más energética (rayos UV de alta frecuencia, rayos X y rayos Gamma) lo que provoca que las pocas moléculas de aire residual que permanecen, se descompongan en radicales libres (oxígeno atómico), iones y electrones. Este fenómeno dota a esta capa del nombre de ionosfera (Marcos-Garrachón, 2024).

Exoesfera Se extiende desde los 500km hasta los 10000km y actúa como frontera con el espacio exterior. Tiene una densidad de gases muy baja que se acerca al casi vacío del espacio exterior.

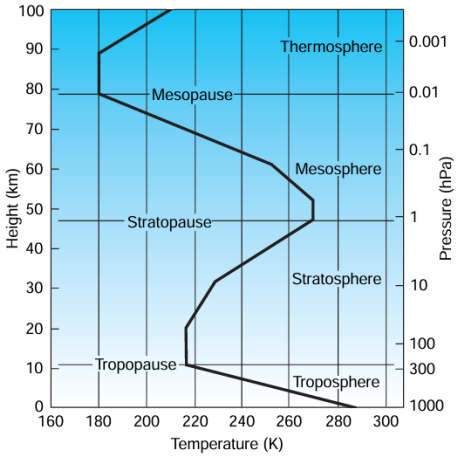

Figura A.1: Perfil vertical de temperatura típico de latitudes medias (Wallace *et al.*, 2006

A.2. El ciclo hidrológico o ciclo del agua

Si aproximamos la Tierra y la atmósfera como un sistema cerrado, en relación con el agua en nuestro planeta, está probado mediante numerosos análisis del nivel del agua en reservas, que la cantidad neta de agua presente en este se mantiene constante a lo largo del tiempo. Obviamente esto es una aproximación y en la realidad se observan pequeñas pérdidas. No obstante, el porqué prácticamente se conserva todo el agua dentro de nuestro sistema, está intrínsecamente relacionado con la creación de las nubes y el perfil vertical de temperaturas de la troposfera. Aquellos lugares del planeta en los que el agua se almacena o por los que pasa en algún momento de su ruta por el sistema Tierra-atmósfera se los denomina reservas y el conjunto de las diferentes reservas acuáticas en nuestro planeta es lo que se denomina la hidrosfera. La hidrosfera se compone de: glaciares, mares, océanos, ríos, lagos, agua subterránea, atmósfera y biosfera.

El *Atlas de Nubes y Meteoros* (2013) de Jose A. Gallego y Jose A. Quirantes describe el ciclo del agua de una forma más cercana a posibles lectores no relacionados con él mediante la descripción del recorrido de una gota de agua: la gota divide su peripecia en tres fases principales. En primer lugar, parte de la superficie terrestre en donde se encuentra almacenada en estado sólido o líquido en alguna de las reservas de la hidrosfera. Pasa mediante evaporación a la atmósfera como vapor de agua, aunque también puede pasar directamente desde el estado sólido mediante sublimación, o por el proceso de transpiración de los seres vivos. En segundo lugar, el vapor de agua en la atmósfera debido a los cambios de presión y temperatura, y con la ayuda de los núcleos de condensación, se condensa en pequeñas gotas o si las condiciones son propicias en pequeños cristales de hielo, en suspensión, formando así las nubes. En tercer lugar, cuando las gotas o cristales crecen y se vuelven lo suficientemente pesados, caen sobre la superficie terrestre en forma de precipitación. Una vez cae sobre la superficie terrestre de nuevo, la gota podría ser transportada por vientos intensos y volver a evaporarse sin tan siquiera llegar a la Tierra. Podría caer sobre un océano, ser atrapada por fuertes corrientes verticales y pasar así miles de años en la profundidad del océano hasta volver a evaporarse; o podría caer en un desierto donde sería rápidamente absorbida y pasaría a formar parte de la reserva de agua subterránea. Se podrían nombrar innumerables caminos que podría seguir la gota, sin embargo, lo relevante reside en el balance constante entre la tasa de evaporación y la tasa de precipitación (Wallace *et al.*, 2006, chapter 2):

$$P - E + Tr = 0$$

Donde, P es la tasa de precipitación, E la tasa de evaporación y Tr es un término de transporte del vapor de agua horizontal por los vientos.

Al discutir los diversos intercambios entre reservas del ciclo hidrológico, se utiliza el concepto de tiempo de residencia, que se define como la masa de la reserva dividida entre el flujo másico neto de salida de esta. El tiempo de residencia es un indicador de la cantidad de tiempo que pasa una molécula de agua dentro de dicha reserva. Tiempos de residencia largos se asocian a grandes reservas. La mayor reserva de agua actualmente es el manto terrestre, y tiene un tiempo de residencia de 10^{11} años (Wallace *et al.*, 2006, chapter 2). Si, por ejemplo, calculamos el tiempo de residencia del agua en la atmósfera dividiendo la masa estimada de agua en la atmósfera por la tasa media de precipitación, obtenemos un resultado de 10 días, siendo así la atmósfera, en concreto la troposfera, de nuevo la reserva más dinámica de la hidrosfera.

A.3. Mecanismos de saturación

A continuación, abordaremos los mecanismos de saturación más importantes, que pueden verse en mayor profundidad en *Atlas de Nubes y Meteoros* de Jose A. Gallego y Jose A. Quirantes.

Frentes

Se define frente como la frontera entre dos masas de aire con diferentes características de humedad y temperatura, y frontogénesis como el proceso mediante el cual un frente se genera. No resulta fácil la distinción de una masa de aire por su humedad o temperatura, ya que estas son propiedades que suelen fluctuar o variar fácilmente, por ello se han definido magnitudes quasi-invariantes en muchos de estos procesos de frontogénesis para poder seguir la evolución de las diferentes masas de aire. Estas magnitudes son temperaturas definidas como: la temperatura potencial equivalente y la temperatura potencial del termómetro húmedo, que nos ayudan a diferenciar entre masas de aire frías y cálidas (relativamente).

Se habla de frente cálido cuando una masa de aire relativamente cálido que avanza se encuentra con aire más frío. Al chocar, el aire cálido mucho menos denso, se ve forzado a ascender sobre el aire frío, que actúa a forma de rampa para el cálido. Se conoce a dicha rampa como superficie frontal, y el ascenso forzado sobre ella constituye uno de los mecanismos de formación de nubes. Al ascender, la masa de aire disminuye su presión (tiene menos masa de aire encima y por tanto, menos presión atmosférica.

Por tanto, al disminuir la presión el aire y el vapor de agua se enfrían, lo que da lugar a la saturación del aire a diferentes alturas dependiendo de su humedad. La nubosidad generada por los frentes cálidos se caracteriza por nubes estratiformes, de escaso desarrollo vertical, como los *altostratus* o los *nimbostratus*.

Un frente frío en su lugar se define como una masa de aire frío que avanza hacia una masa de aire cálido mucho más ligera, por lo que al entrar en contacto el aire frío toma una forma de cuña y se desplaza por debajo de la masa cálida. El aire cálido de nuevo se ve forzado a ascender, lo que desencadena el mismo mecanismo de disminución de presión y saturación del aire. Ambos procesos se diferencian fundamentalmente en que los frentes fríos fuerzan a un ascenso mucho más vertical de la masa cálida, lo que da lugar a la formación de nubes de desarrollo vertical como *cumulus* o *cumulonimbus*.

Radiación y convección

La atmósfera es muy transparente a la radiación solar de onda corta que recibe el Sol, por lo tanto, la troposfera se calienta principalmente por radiación de la superficie terrestre. La corteza terrestre absorbe la radiación solar de onda corta y emite radiación saliente de onda larga o infrarrojo térmico. Esta radiación sí que es absorbida por parte de la atmósfera y es el mecanismo que más contribuye al calentamiento de la parte inferior de la troposfera.

El calor se transfiere mediante conducción, convección y radiación. Sobre la radiación ya hemos hablado, y el siguiente mecanismo en entrar en juego es la convección. La convección es uno de los mecanismos que utiliza la atmósfera para mantener el balance energético, transportando calor entre zonas con temperaturas diferentes. La dirección de este intercambio de energía nos dice el principio cero de la termodinámica que siempre será de zonas más calientes a zonas más frías.

La conducción no tiene relevancia prácticamente ya que la baja conductividad térmica del aire hace que casi no se transfiera energía mediante este mecanismo (en comparación con los otros).

Así pues, para que el mecanismo de convección se ponga en marcha, necesitamos masas de aire que estén más calientes que su entorno. Por ejemplo, la radiación absorbida por la superficie terrestre no será la misma que la absorbida por el océano. Por ejemplo, si tenemos una isla, se puede crear una capa de aire superficial notablemente más caliente generada por la emisión del terreno de la isla. Mientras que esa misma capa superficial para el mar, no estará tan caliente. Ya que el mar al tener una capacidad calorífica menor no es tan susceptible a cambios en su temperatura, y por tanto emitirá menos radiación térmica. Por convección, se crea una masa de aire cálido por encima de la isla, rodeada de aire más frío a la misma altura. Debido a la menor densidad del aire caliente, la masa de aire caliente tiende a ascender, creándose así corrientes ascendentes. La altura que alcancen durante el proceso dependerá de la inestabilidad de las capas de la troposfera y de cuanto tarde el equilibrio en alcanzarse. El

proceso mediante el que las masas de aire caliente ascienden, es un proceso adiabático, por lo que toda la variación de temperatura se deberá a la variación de la presión. A una cierta altura, el aire ascendente comenzará a saturarse y a condensarse; a esta altura se la denomina nivel de condensación convectivo.

Este proceso se asimila en gran medida a los frentes fríos y las nubes resultantes de ambos son las mismas: *cúmulus* y *cumulonimbus*. No obstante, el mecanismo por el cual las nubes creadas por convección siguen desarrollándose verticalmente es que cuando las primeras gotas líquidas comienzan a formarse, el cambio de estado gaseoso a estado líquido, libera calor latente, que ralentiza el proceso de enfriamiento. Las moléculas en estado gaseoso continuarán subiendo hasta que todas se condensan y forman la estructura de la nube con un desarrollo vertical. Es por tanto esta dilatación temporal en el mecanismo de formación de las nubes por convección lo que produce su estructura alargada, mientras que, en los frentes fríos, el perfil vertical de las nubes se debe más a la interacción con la masa de aire caliente, y la obligación de este de ascender. Sin embargo, lo más importante que cabe decir es que, ninguno de estos procesos en la naturaleza se da de forma aislada, si no que interactúan unos con otros y dando como resultado final la formación de las nubes.

Convergencia de viento

La masa de aire atmosférico se reparte de forma desigual en toda su vertical, lo que produce zonas de altas y bajas presiones y origina movimientos de masas de aire, que es lo que se conoce como viento, que va de zonas de alta presión a zonas de baja presión. Como hemos dicho todos los mecanismos de saturación están saturados y aquí podemos ver un ejemplo más, ya que la principal causa del reparto desigual de masa atmosférico es la cantidad de radicación solar que reciben diferentes zonas de la Tierra, así como diferencias en la radiación que emiten de vuelta (pero principalmente la que reciben). Por lo que, cuestiones como la latitud o el tipo del terreno sobre el que se encuentre la masa de aire son diferenciables a la hora de determinar su comportamiento.

Se denomina convergencia horizontal, al movimiento horizontal de masas de aire que convergen en un punto. Como la acumulación de masa no es posible, esto da lugar a corrientes verticales con una intensidad y velocidad proporcionales a los de la convergencia que los induce. Este mecanismo de ascenso esta estrechamente ligado como hemos visto con la convección. De hecho, si la masa de aire tras la convergencia alcanza un punto de inestabilidad, tenemos de nuevo, que seguirá subiendo esta vez puramente por convección. De cualquier modo, tanto si tenemos una convergencia lo suficientemente potente para saturar el aire, o si la masa alcanza un punto inestable y sigue subiendo por convección, tenemos que las nubes formadas son de nuevo *cumulus* o *cumulonimbus*.

También puede darse, que en la convergencia estemos en un punto de estabilidad atmosférica en la vertical. En este caso, si la masa de aire tiene la suficiente humedad, la convergencia puede convertirse en una zona de convergencia de humedad. Se puede así saturar la masa por un aumento en la humedad, formándose así nubes como los *stratus* o *stratocumulus*.

Ascensos orográficos

Cuando un flujo de aire se encuentra con un obstáculo montañoso, o bien lo rodea, o bien asciende para superarlo. En caso de ascender para superarlo, se enfriará adiabáticamente, pudiendo llegar a la saturación y a la condensación. Este es el efecto Foehn y puede ser a la inversa, cuando una nube desciende por la ladera de una montaña y se calienta. Los sistemas

montañosos conforman pues un mecanismo de formación de nubes o actúan como detonadores de otros mecanismos de formación.

Los otros mecanismos también se relacionan con la orografía, pudiendo las laderas de las montañas recibir más o menos radiación en función de su orientación, provocando así ascensos por convección, o pudiendo dar lugar a convergencias de aire, fruto de canales orográficos que conducen el flujo de aire hacia las convergencias.

Enfriamiento por irradiación

Durante la noche, sobre todo en invierno, en situaciones de estabilidad atmosférica, es decir, sin nubes o viento, sin la radiación solar se produce un intenso enfriamiento de la superficie, que por conducción se sucede a una fina capa de aire contigua (ya que el aire es muy mal conductor). Este enfriamiento suele quedar restringido a unos metros de altura de la superficie terrestre y da lugar a una inversión en el gradiente de temperatura en dicho espacio. Si la temperatura del aire desciende por debajo del punto de rocío, temperatura a la cual el agua empieza a condensarse o sublimarse, comenzará a depositarse rocío o escarcha sobre los objetos de la superficie terrestre. Si el enfriamiento continua, el viento es débil y existen los suficientes núcleos de condensación; la saturación del vapor de agua ya no solo afectara a la capa más próxima a la tierra si no que se extenderá hasta llegar incluso a cientos de metros, formando lo que se denomina niebla.

A.4. Núcleos de condensación

Hemos hablado de los diferentes mecanismos por los cuales se satura el aire con vapor de agua y eso da lugar a la condensación, sin embargo, ¿qué es la saturación y como se condensa el agua? Para dar respuesta a nuestra pregunta debemos recurrir a los recientemente mencionados núcleos de condensación o congelación. La principal función de estos núcleos es encargarse de vencer las fuerzas de tensión superficial, que impiden que las moléculas de vapor de agua formen gotas o cristales por si solas. Sin su ayuda, los procesos físicos observados en la formación de las nubes raramente se darían.

El aire está conformado por una mezcla de aire seco y vapor de agua. Se dice por tanto que el aire tiene una humedad, definida como (Kshudiram, 2008, chapter 5):

"La densidad o la masa de vapor de agua por unidad de volumen en un volumen de aire húmedo, medida en unidades de g por cm^3 o kg por m^3".

Definimos también por su utilidad en el futuro el *humidity mixing ratio* (h.m.r.) como:

La cantidad de vapor de agua en gramos mezclada con un kg de aire seco. Por tanto, si tenemos que m_v gramos de vapor de agua se mezclan con m_d kg de aire seco, su hmr viene dado por m_v/m_d.

La condición de saturación del aire se da cuando la presión parcial del vapor de agua en la mezcla de gases alcanza la presión de saturación ($e = e_s$). Dicho esto, la masa de aire húmedo no-saturado ascenderá por uno de los mecanismos comentados anteriormente, disminuyendo la presión y produciéndose un proceso pseudo adiabático de expansión. Su ascenso se ve ralentizado en el momento en que llega al punto de expansión y presión en el cual el aire se encuentra a una temperatura tal que, la presión de vapor de saturación disminuye más de lo que lo hace la presión del aire, con lo que se consigue que se de la condición de saturación del aire.

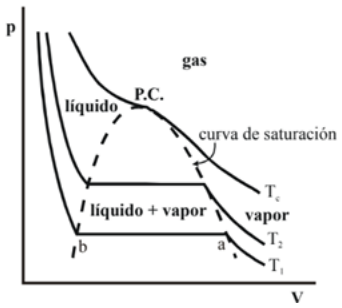

Figura A.2: Diagrama de isotermas en el intervalo de la curva de saturación. Dado que en el cambio de fase la temperatura se mantiene constante, podemos modelizar el proceso mediante este diagrama (Casanova, 2018).

En la Figura A.2 podemos observar dicho punto *a*, que tendrá lugar a una determinada temperatura, volumen y presión. En dicho momento, el enfriamiento se ralentiza ya que a temperatura y presión constante se produce un cambio de fase hasta tener líquido saturado en el punto *b*. En este proceso se libera el calor latente de cambio de fase. Sin embargo, con todo, el proceso seguiría siendo reversible y adiabático si todos los productos del cambio de fase y el calor liberado se mantuviesen en el sistema. Por desgracia, cuando algunos de los productos se pierden por precipitación, estos se llevan además consigo una cantidad de calor, por lo que el proceso deja de ser adiabático y reversible. Dado que la mayor parte de los productos no se precipitan, sino que se mantienen en suspensión formando una nube, podemos depreciar esta pérdida de calor y tomar el proceso como pseudo-adiabático (Jacobson, 2012, chapter 16).

Podemos definir ahora como ratio de lapso pseudo-adiabático la tasa de cambio de la temperatura de una masa de aire en ascenso cuando experimenta una expansión pseudo-adiabática, es decir, en condiciones en las que se permite la saturación y, por tanto, la emisión de calor latente al condensarse. Esta tasa ($6°C/km$) suele ser menor que el caso del lapso adiabático del aire seco, que es de $9{,}8°C/km$.

Una vez llegamos al punto de saturación, vamos a analizar el por qué la condensación no ocurre, y en la mayoría de los casos se necesita de la presencia de núcleos de condensación o aire sobresaturado, que se dice cuando el aire contiene más vapor de agua de lo que puede disolver/mezclar a una temperatura o presión específicas. Podemos anticipar que esto ocurre por la imposibilidad del vapor de agua saturado de condensarse y pasar a formar gotas de agua, debido a la barrera potencial que supone la energía de superficie, más comúnmente conocida como tensión superficial (se puede demostrar que el valor numérico de ambas es el mismo. La tensión superficial actúa como retenedor de fase para el vapor de agua, dándose así la sobresaturación.

Para ver esto en más detalle vamos a ponernos en el caso de la formación de una gota de agua, sin ayuda de otras partículas (núcleos de condensación). A este proceso se lo conoce como Nucleación Homogénea de Condensación [1](Wallace *et al.*, 2006). Partimos de considerar una pequeña gota de agua formada mediante colisiones de vapor de agua sobresaturado a presión y temperaturas constantes, con volumen V y superficie A. Si μ_l y μ_v son las energías libres de Gibbs por molécula en fase líquida y gaseosa respectivamente y n es la densidad

[1]La Nucleación Homogénea se refiere a la condensación de vapor de agua en gotas de agua, sin ayuda de elementos externos, mientras que la nucleación heterogénea se refiere a la condensación del vapor de agua con ayuda de núcleos de condensación (Yau & Rogers, 1989).

volumínica molecular, el descenso en la energía libre de Gibbs en el sistema es: $n \cdot V \cdot (\mu_v - \mu_l)$. Además, en la creación de la superficie de la gota se realiza un trabajo que se puede expresar como el área por la tensión superficial (σ). El incremento neto de la energía en la formación de la gota es:

$$\Delta E = A\sigma - nV(\mu_v - \mu_l) \tag{A.1}$$

Donde se puede demostrar que [2]:

$$\mu_v - \mu_l = kTln(\frac{e}{e_s}) \tag{A.2}$$

Donde e y T son la presión de vapor y la temperatura del sistema, ‖ es la constante de Boltzman y e_s es la presión de saturación del vapor de agua sobre una superficie plana. Reescribiendo:

$$\Delta E = A\sigma - nVkTln(\frac{e}{e_s}) \tag{A.3}$$

Si la ponemos en función del radio de la gota:

$$\Delta E = A\sigma - \frac{4}{3}\pi R^3 nVkTln(\frac{e}{e_s}) \tag{A.4}$$

Conviene aclarar que: un proceso es espontáneo si la variación de la energía libre de Gibbs ΔG es negativa. En ese caso, el proceso puede ocurrir sin la necesidad de aporte de energía externa.

En condiciones subsaturadas $ln(e/e_s)$ es negativo y por tanto, el incremento de energía es positivo e incrementa con r. Lo que nos dice, que claramente la creación de gotas de agua en condiciones subsaturadas no se ve nada favorecida, aunque estadísticamente no es imposible, y de hecho, algunas gotas se forman de esta manera, evaporándose casi instantáneamente tras su formación.

En condiciones sobresaturadas $ln(e/e_s)$ es positivo. En este caso, el incremento de energía puede ser negativo o positivo dependiendo del radio como podemos observar en la Figura A.3.

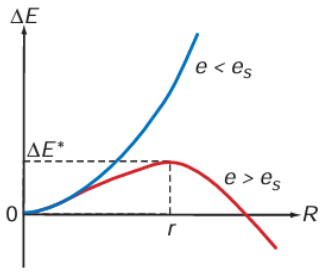

Figura A.3: ΔG de un sistema por la formación de una gota de agua de radio R con una presión de vapor e, donde e_s, es la presión de saturación de vapor con respecto a una superficie plana de agua a la temperatura del sistema (Wallace *et al.* 2006).

Al inicio, el incremento de energía es positivo e incrementa con el radio hasta llegar al punto $r = R$, en donde la pendiente cambia y empieza a hacerse negativo y por tanto el proceso de formación de gotas comienza a ser espontáneo. Si $r < R$, entonces las gotas se evaporarán rápidamente como en el caso subsaturado; sin embargo, si mediante colisiones se consiguen

[2]Ver Capítulo 2 en *Fundamentals of Physics and Chemistry of the Atmosphere (2016)*, de Guido Visconti.

formar gotas con un $r > R$, entonces estas seguirán creciendo mediante condensación del vapor, ya que esto producirá un descenso en la energía libre de Gibbs.Podemos obtener el máximo de la función que equivale al punto $r = R$:

$$r = \frac{2\sigma}{nkTln(\frac{e}{e_s})} \tag{A.5}$$

Esta ecuación se denomina ecuación de Kelvin, ya que Lord Kelvin fue el primero en derivarla. Al derivar esta ecuación hemos realizado una aproximación fundamental al considerar e_s como la presión de saturación del vapor de agua sobre una superficie plana, ya que nuestras gotas no son planas, sino que son, en una buena aproximación, esféricas.

Fue Lord Kelvin (1870) de nuevo el que demostró que la presión de saturación de los líquidos sobre una superficie curva como la de una pequeña gota de radio r, era considerablemente mayor que para una superficie plana. La relación que desarrolló es [3]:

$$ln(\frac{e_r}{e_s}) = \frac{2\sigma}{rnkT} \tag{A.6}$$

Donde e_r y e_s representan las presiones de vapor sobre una superficie esférica de radio r y una superficie plana respectivamente. Lo que, para una gota de agua a $10°C$, nos da los siguientes resultados expuestos en el Tabla A.1:

r(cm)	$2 \cdot 10^{-8}$	10^{-7}	10^{-6}	10^{-5}	10^{-4}
e_r/e_s	316.2	3.162	1.127	1.012	1.001

Tabla A.1: Fracción de e_r/e_s para una gota de agua a 10^{o}C (Kshudiram, 2008).

La molécula de H_2O tiene un radio de $2 \cdot 10^{-8}$ cm. Normalmente, en el proceso de enfriamiento, se deben agrupar casi 100 moléculas de agua para formar una pequeña gota de radio aproximadamente 10^{-7} cm. Según la ecuación (A.6) para una cierta temperatura (T) y radio (r) a densidad molecular constante, la tensión superficial depende directamente del cociente e_r/e_s. Para una gota de radio $10^{-7}cm$ necesitaríamos tener una presión de vapor 3.1 veces la presión para una superficie plana e_s (ver Tabla A.1). Esto deriva en una mayor tensión superficial, es decir barrera energética que superar. Lo que nos lleva a la necesidad de una mayor sobresaturación de vapor de agua en el aire si queremos formar una gota esférica con e_s. En ausencia de tal sobresaturación, la gota se evaporaría tan pronto como se forma.

Si hay una partícula de polvo con un radio de $10^{-6}cm$ presente, las moléculas de agua que se depositan sobre ella forman una gota con un radio de $10^{-6}cm$, y ahora el cociente entre e_r/e_s es solo 1.12, es decir, que la energía necesaria para superar la tensión superficial es aproximadamente la misma que para una superficie plana. Por lo tanto, las gotas formadas por la deposición de moléculas de agua sobre partículas de polvo son mucho más probables debido a su eficiencia en la formación. Los experimentos de laboratorio demuestran que debe existir algún tipo de núcleo si se quiere que el vapor de agua enfriado forme niebla o nubes bajo condiciones atmosféricas (Kshudiram, 2008, chapter 5).

Este es el papel de los núcleos de condensación. Algunos son higroscópicos, es decir, que atraen el agua hacia ellos, en cuyo caso la sobresaturación que se necesita para las gotas es mucho menor, y otros están cargados, lo que produce una reducción en la presión de saturación sobre una superficie esférica que se necesita.

[3]Para ver en mayor profundidad el desarrollo matemático así como la dependencia con la carga ver Capítulo 5 de *The Earth´s Atmosphere. Its physics and dynamics* (2008) de Saha Kshudiram.

Apéndice B

Backpropagation

A pesar del número exponencial de caminos y parámetros a optimizar, hoy en día se puede llevar a cabo el algoritmo de backpropagation o retropropagación gracias a la programación dinámica. Podemos diferenciar dos fases[1] (Bosch *et al.*, 2019):

B.1. Forward Pass

Se alimenta la red con los inputs y se calcula la salida haciendo uso del set de pesos y sesgos que posee en dicho momento la red. Posteriormente se calcula el error de nuestra salida respecto de la salida real.
Vamos ahora a desarrollar el algoritmo de backpropagation para una red neuronal de L capas, pero antes de nada veamos la notación que vamos a usar:

- n_l es el número de neuronas en la capa l y n_0 es la dimensión de los datos de entrada.
- $X \in M_{n_0 \times m}(\mathbb{R})$ es la matriz de datos de entrada. Cada instancia o ejemplo está representado en una columna y los atributos en una fila.
- $W^{[l]} \in M_{n_l \times n_{l-1}}(\mathbb{R})$ es la matriz de pesos que conecta la capa *l-1* con la capa *l*.
- w^l_{jk} elemento de $w^{[l]}$bde la fila j y la columna k. Representa el peso que regula la conexión entre la neurona j de la capa l y la neurona k de la capa *l-1*.
- $b^{[l]} \in M_{n_l \times 1}(\mathbb{R})$ denota los bias o sesgos de la capa *l*. b^l_j es el sesgo o bias de la neurona j en la capa *l*.
- $z^{[l]} \in M_{n_l \times 1}(\mathbb{R})$ denota la combinación lineal de entrada a la capa l con los parámetros $W^{[l]}$ y $b[l]$.
- $g : \mathbb{R} \to \mathbb{R}$ función no lineal de activación. Si $M \in M_{c \times d}(\mathbb{R}) \Rightarrow g(M) \in M_{c \times d}(\mathbb{R})$. Mantiene las dimensiones del objeto sobre el que se aplica. Actúa componente a componente.
- $a^{[l]} \in M_{n_l \times 1}(\mathbb{R})$ denota el vector de salida de la capa l de una red neuronal. $a^{[l]}$ Denota el vector de salida de la red y $a^{[0]}$ denota el vector de atributos de entrada a la red neuronal.

Consideramos ahora m instancias o ejemplos con no atributos, es decir, $X \in M_{n_0 \times m}(\mathbb{R})$.Vamos a expresar el número de ejemplo o instancia entre paréntesis como un superíndice: $x^{[l](1)}$ es el

[1]Para la realización de este apéndice se ha usado como referencia el Apéndice B de Bosch *et al.* (2019), que lleva el nombre de Backpropagation. Se ha cambiado la notación utilizada para dar una visión más general.

vector columna que representa los n_0 atributos de entrada del primer ejemplo o instancia.
Calculamos la salida de la red para cada instancia o ejemplo mediante las ecuaciones:

$$z_j^{[l](s)} = \sum_{k=1}^{n_l-1} (w_{jk}^{[l](s)} a_k^{[l-1](s)}) + b_j^{[l](s)} \tag{B.1}$$

$$a_j^{[l](s)} = g(z_j^{[l](s)}) \tag{B.2}$$

Las ecuaciones están representadas para el s-ejemplo en la l-capa, pero se pueden dar de forma matricial como:

$$Z^{[l]} = W^{[l]} * A^{[l]} + B^{[l]} \tag{B.3}$$

$$A^{[l]} = g(Z^{[l]}) \tag{B.4}$$

Donde: $Z^{[l]} = (z^{[l](1)}, z^{[l](2)}, \ldots, z^{[l](s)}, \ldots, z^{[l](m)})$, $A^{[l]} = (a^{[l](1)}, a^{[l](2)}, \ldots, a^{[l](s)}, \ldots, a^{[l](m)})$ con $A^{[0]} = X$ y $*$ representa el producto matricial.

Aplicando dichas ecuaciones sobre todas las capas, y sobre todos los ejemplos, obtenemos la salida de la red:

$$A^{[L]} = (a^{[L](1)}, a^{[L](2)}, \ldots, a^{[L](s)}, \ldots, a^{[L](m)})$$

Donde se denota por $[L]$ mayúscula a la última capa de la red, es decir, que $[l]$ se encuentra comprendido entre: $1 < l < L$. Posteriormente analizamos la salida de la red con la función de coste definida ($\mathcal{L}$) y con las salidas reales $y^{(s)}$:

$$\mathcal{L}(Y, A^{[L]}) = (\mathcal{L}(y^{(1)}, a^{[L](1)}), \mathcal{L}(y^{(2)}, a^{[L](2)}), \ldots, \mathcal{L}(y^{(s)}, a^{[L](s)}), \ldots, \mathcal{L}(y^{(m)}, a^{[L](m)})$$

Finalmente, dado que derivar es una aplicación lineal, es decir, cumple las propiedades de homogeneidad y aditividad, podemos calcular la función de coste para cada ejemplo por separado y luego realizar la media. Definimos la función de coste global ($\mathcal{J}$) como:

$$\mathcal{J} = \frac{1}{m} \cdot \sum_{s=1}^{m} \mathcal{L}(y^{(s)}, a^{[L](s)})$$

Esto es lo que se conoce como el forward pass, en el que partiendo de los atributos de entrada llegamos a la salida y al cálculo del error cometido por la red.

B.2. Backpropagation

vamos a hallar el gradiente de la función de coste global respecto de los parámetros que queremos optimizar, para hallar el mínimo global de dicha función mediante el algoritmo del descenso del gradiente.
En el *Forward Pass* hemos hallado $\mathcal{J}(Y, A^{[L]})$, sin embargo sabemos que podemos derivar la dependencia explícita con respecto de $A^{[L]}$, a una dependencia respecto de los pesos y sesgos $\mathcal{J}(W^{[L]}, B^{[L]})$.
Para ello vamos a aplicar la *regla de la cadena multivariable*:
Sean f y g tal que:

$$f : \mathbb{R} \to \mathbb{R} \qquad\qquad g : \mathbb{R} \to \mathbb{R}$$

$$x \to f(x) \qquad\qquad y \to g(y)$$

La regla de la cadena nos dice que la derivada de la composición g(f(x)) respecto de x, viene dada por el producto:

$$\frac{\partial(g \circ f)}{\partial x} = \frac{\partial g}{\partial f} \cdot \frac{\partial f}{\partial x} \tag{B.5}$$

Si ahora tuviesemos que f y g son funciones en varias variables:

$$f : \mathbb{R} \to \mathbb{R} \qquad\qquad g : \mathbb{R} \to \mathbb{R}$$

$$x \to (f_1(x), \ldots, f_d(x)) \qquad\qquad (y_1, \ldots, y_d) \to (g_1(y), \ldots, g_d(y))$$

La regla de la cadena multivariable nos dice que la derivada de la composición g(f(x)) respecto de x, viene dada por la fórmula:

$$\frac{\partial(g \circ f)}{\partial x} = \sum_{c=1}^{d} \frac{\partial g}{\partial f_c} \cdot \frac{\partial f_c}{\partial x} \tag{B.6}$$

Utilizando ahora la regla de la cadena multivariable podemos calcular la derivada parcial de la función de coste respecto de cualquier parámetro de la red. En general la dependencias las podemos descomponer como:

$$\mathbb{R} \to \mathbb{R} \to \mathbb{R} \to \mathbb{R} \qquad\qquad \mathbb{R} \to \mathbb{R} \to \mathbb{R} \to \mathbb{R}$$

$$w_{jk}^{[l](s)} \to z_j^{[l](s)} \to a_j^{[l](s)} \to \mathcal{L}^{(s)} \qquad\qquad b_j^{[l](s)} \to z_j^{[l](s)} \to a_j^{[l](s)} \to \mathcal{L}^{(s)}$$

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^{[l](s)}} = \frac{\partial \mathcal{L}^{(s)}}{\partial a_j^{[l](s)}} \cdot \frac{\partial a_j^{[l](s)}}{\partial z_j^{[l](s)}} \cdot \frac{\partial z_j^{[l](s)}}{\partial w_{jk}^{[l](s)}} \tag{B.7}$$

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^{[l](s)}} = \frac{\partial \mathcal{L}^{(s)}}{\partial a_j^{[l](s)}} \cdot \frac{\partial a_j^{[l](s)}}{\partial z_j^{[l](s)}} \cdot \frac{\partial z_j^{[l](s)}}{\partial b_j^{[l](s)}} \tag{B.8}$$

Donde es fácil ver que:

$$\frac{\partial z_j^{[l](s)}}{\partial w_{jk}^{[l](s)}} = \frac{\partial(w_{jk}^{[l](s)} a_k^{[l-1](s)} + b_j^{[l](s)})}{\partial w_{jk}^{[l](s)}} = a_k^{[l-1](s)} \tag{B.9}$$

$$\frac{\partial z_j^{[l](s)}}{\partial w_{jk}^{[l](s)}} = \frac{\partial(w_{jk}^{[l](s)} a_k^{[l-1](s)} + b_j^{[l](s)})}{\partial b_j^{[l](s)}} = 1 \tag{B.10}$$

$$\frac{\partial a_k^{[l](s)}}{\partial z_j^{[l](s)}} = \frac{\partial g(z_j^{[l](s)})}{\partial z_j^{[l](s)}} = g'(z_j^{[l](s)}) \tag{B.11}$$

Por lo tanto, podemos reescribir sustituyendo las expresiones (B.9), (B.10) y (B.11) en (B.7) y (B.8):

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^{[l](s)}} = \frac{\partial \mathcal{L}^{(s)}}{\partial a_j^{[l](s)}} \cdot g'(z_j^{[l](s)}) \cdot a_k^{[l-1](s)} \tag{B.12}$$

$$\frac{\partial \mathcal{L}}{\partial w_{jk}^{[l](s)}} = \frac{\partial \mathcal{L}^{(s)}}{\partial a_j^{[l](s)}} \cdot g'(z_j^{[l](s)}) \tag{B.13}$$

En notación matricial e introduciendo de nuevo la función de coste global ($\mathcal{J}$):

$$\frac{\partial \mathcal{J}}{\partial W^{[l]}} = \frac{1}{m} \cdot \frac{\partial \mathcal{L}}{\partial A^{[l]}} \odot g'(Z^{[l]}) * (A^{[l-1]})^T \tag{B.14}$$

$$frac \partial \mathcal{J} \partial B^{[l]} = \frac{1}{m} \cdot \sum_{s=1}^{m} \frac{\partial \mathcal{L}^{(s)}}{\partial a^{[l](s)}} \cdot g'(Z^{[l](s)}) \tag{B.15}$$

Donde: $\odot$ es el producto componente a componente, $\frac{\partial \mathcal{L}}{\partial W^{[l]}} \in M_{n_l \times n_{l-1}}$, $\frac{\partial \mathcal{J}}{\partial A^{[l]}} \in M_{n_l \times m}$, $g'(Z^{[l]}) \in M_{n_l \times m}$, $(A^{[l-1]})^T \in M_{m \times n_{l-1}}$ y $\frac{\partial \mathcal{J}}{\partial B^{[l]}} \in M_{n_l \times 1}$.

Manejar las dimensiones matriciales de los elementos, no es una tarea fácil, sin embargo, si fundamental, para entender el proceso que sigue el algoritmo.
Por último, nos queda ver la derivada $\frac{\partial \mathcal{L}}{\partial A^{[l]}}$. Para ello debemos diferenciar dos casos:

Caso 1: $\frac{\partial \mathcal{L}}{\partial A^{[L]}}$ En este caso, dado que conocemos la forma explícita de la dependencia de $\mathcal{L}$ con $A^{[L]}$ podemos hallar la derivada mediante un cálculo directo. Por ejemplo en el caso de haber definido como función de coste la función Error Cuadrático Medio (MSE) y tener una salida binaria:

$$\frac{\partial \mathcal{L}}{\partial a^{[L](s)}} = -(Y^{(s)} - a^{[L](s)}) \tag{B.16}$$

En este caso tendriamos que $A^{[L]}$ es un vector fila de ceros y unos, donde cada cero y cada uno es el $a^{[L](s)}$ correspondiente a cada columna.

Caso 2: $\frac{\partial \mathcal{L}}{\partial A^{[l]}}$ **con** $\mathbf{l < L}$ Este caso es más complejo que el anterior ya que la activación de una neurona de la capa l afecta a todas las neuronas de la capa l+1. Como estamos propagando el error hacia atrás, es decir desde la última capa hacia la primera, decir que afecta a todas las neuronas de la capa l+1, es lo mismo que decir que: $a^{[l](s)} = f(a_1^{[l+1](s)}, \ldots, a_i^{[l+1](s)}, \ldots, a_{n_{l+1}}^{[l+1](s)})$. Podemos expresar $a^{[l](s)}$ como función de las salidas de la capa siguiente $a_i^{[l+1](s)}$, donde el subíndice i índica la salida de la neurona i de la capa l+1, y n_{l+1} el número de neuronas en la capa l+1. Podemos a su vez descomponer la dependencia pues en las $z_i^{[l+1](s)}$:

$$\mathbb{R} \to \mathbb{R} \to \mathbb{R} \to \mathbb{R}$$
$$a_j^{[l](s)} \to (z_1^{[l+1](s)}, \ldots, z_{n_{l+1}}^{[l+1](s)}) \to (a_i^{[l+1](s)}, \ldots, a_{n_{l+1}}^{[l+1](s)}) \to \mathcal{L}^{(s)}$$

Por tanto aplicando ahora la regla de la cadena multivariable:

$$\frac{\partial \mathcal{L}}{\partial a_j^{[l](s)}} = \sum_{c=1}^{n_{l+1}} \frac{\partial \mathcal{L}^{(s)}}{\partial a_c^{[l+1](s)}} \cdot \frac{\partial a_c^{[l+1](s)}}{\partial z_c^{[l+1](s)}} \cdot \frac{\partial z_c^{[l+1](s)}}{\partial a_j^{[l](s)}} \tag{B.17}$$

Por supuesto, $a_c^{[l+1](s)}$ y $z_c^{[l+1](s)}$ tienen el mismo subíndice c, ya que los valores de activación de cada neurona solo afectan a la salida de dicha neurona. De nuevo si recordamos:

$$\frac{\partial z_c^{[l+1](s)}}{\partial a_j^{[l](s)}} = \frac{\partial (w_{cj}^{[l+1](s)} a_j^{[l](s)} + b_j^{[l+1](s)})}{\partial a_j^{[l](s)}} = w_{cj}^{[l+1](s)} \tag{B.18}$$

$$\frac{\partial a_c^{[l+1](s)}}{\partial z_c^{[l+1](s)}} = \frac{\partial g(z_c^{[l+1](s)})}{\partial z_c^{[l+1](s)}} = g'(z_c^{[l+1](s)}) \tag{B.19}$$

Y por tanto nuestra ecuación toma la forma final:

$$\frac{\partial \mathcal{L}}{\partial a_j^{[l](s)}} = \sum_{c=1}^{n_{l+1}} \frac{\partial \mathcal{L}^{(s)}}{\partial a_c^{[l+1](s)}} \cdot g'(z_c^{[l+1](s)}) \cdot w_{cj}^{[l+1](s)} \tag{B.20}$$

Que en forma matricial:

$$\frac{\partial \mathcal{L}}{\partial A^{[l]}} = (W^{[l+1]})^T * (\frac{\partial \mathcal{L}}{\partial A^{[l+1]}} \odot g'(Z^{[l+1]})) \tag{B.21}$$

Donde: $\frac{\partial \mathcal{L}}{\partial A^{[l]}} \in M_{n_l \times m}$, $(W^{[l+1]})^T \in M_{n_l \times n_{l+1}}$, $\frac{\partial \mathcal{L}}{\partial A^{[l+1]}} \in M_{n_{l+1} \times m}$, $g'(Z^{[l+1]}) \in M_{n_{l+1} \times m}$, $\odot$ representa el producto componente a componente y $*$ representa el producto matricial usual

Haciendo uso de esta ecuación podemos hallar por recursividad la derivada $\frac{\partial \mathcal{L}}{\partial a_j^{[l](s)}}$ para todo $c \in [1, \ldots, n_{l+1}]$ partiendo de $\frac{\partial \mathcal{L}}{\partial a_j^{[L](s)}}$, que puede calcularse segun se ha especificado en el caso 1.

Una vez terminado el algoritmo de backpropagation, tenemos los valores del gradiente en función de cada peso y cada sesgo, con lo que podemos aplicar el algoritmo del descenso del gradiente para optimizar en función de cada uno de los parámetros de los cuales tenemos su gradiente. Como vemos a medida que la arquitectura sea más compleja el número de parámetros(que en arquitecturas no muy complejas ya es de alrededor de millones) aumenta de forma considerable.

B.3. Notación complementaria

Por su extendido uso, en caso de que el lector este familiarizado con la notación que los incluye, se introducen los denominados *deltas* (δ). En esta sección no vamos a desarrollar de nuevo el formalismo del algoritmo backpropagation con otra notación, sino que tan solo se van a introducir las fórmulas adquiridas previamente para el caso de que al lector le resulten más familiares de esta forma:

$$W^{[l]} = W^{[l]} - \eta \cdot \frac{\partial \mathcal{J}}{\partial W^{[l]}} = W^{[l]} - \eta \cdot \delta^{[l]} \cdot (A^{[l+1]})^T \tag{B.22}$$

$$B^{[l]} = B^{[l]} - \eta \cdot \frac{\partial \mathcal{J}}{\partial B^{[l]}} = B^{[l]} - \eta \cdot \delta^{[l]} \tag{B.23}$$

De lo que se desprende:

$$\delta^{[l]} = \frac{\partial \mathcal{J}}{\partial W^{[l]}} * ((A^{[l+1]})^T)^{-1} = \frac{1}{m} \cdot \frac{\partial \mathcal{L}}{\partial A^{[l]}} \odot g'(Z^{[l]}) * (A^{[l-1]})^T * ((A^{[l+1]})^T)^{-1} = \frac{1}{m} \cdot \frac{\partial \mathcal{L}}{\partial A^{[l]}} \odot g'(Z^{[l]}) \tag{B.24}$$

$$\delta^{[l]} = \frac{\partial \mathcal{J}}{\partial B^{[l]}} = \frac{\partial \mathcal{J}}{\partial B^{[l]}} = \frac{1}{m} \cdot \sum_{s=1}^{m} \frac{\partial \mathcal{L}^{(s)}}{\partial a^{[l](s)}} \cdot g'(Z^{[l](s)}) \tag{B.25}$$

La relación de recurencia para δ^l puede expresarse como [2]:

$$\delta^l = ((W^{[l+1]})^T * \delta^{l+1}) \odot g'(Z^{[l]})$$

Donde: $\delta^{[l]} \in M_{n_l \times m}$, $(W^{[l+1]})^T \in M_{n_l \times n_{l+1}}$, $\delta^{[l+1]} \in M_{n_{l+1} \times m}$, $g'(Z^{[l]}) \in M_{n_l \times m}$, $\odot$ representa el producto componente a componente y $*$ representa el producto matricial usual

[2]La ecuación de recurrencia de las deltas se ha obtenido sustituyendo las fórmulas para las $\delta^{[l]}$ en la ecuación de recurrencia original (B.21).

Apéndice C

Funciones de activación

Las funciones de activación son una parte fundamental de las redes neuronales, permiten que los modelos aprendan patrones complejos. En lo siguiente, se presentan algunas de las funciones de activación más frecuentes.

- ***Función escalón***:

$$\phi(x) = \begin{cases} 1 & \text{si } x \geq \alpha \\ -1 & \text{si } x \leq \alpha \end{cases} \tag{C.1}$$

  Donde alpha actúa como un nuevo valor umbral de la función de activación. La salida es binaria y se puede establecer como 1,-1 o 1,0. Introduce la no-linealidad en la red mediante el salto entre valores.

  Esta función fue la originalmente usada en el perceptrón, el modelo más simple y el primero de red neuronal. Contaba de una sola neurona con la suma ponderada como función de entrada y la función escalón como función de activación (Bosch *et al.*, 2019).

  El motivo por el que esta función no es utilizada hoy en día, y por lo que el avance tras el perceptrón se paralizo, es que dicha función tiene derivada nula para todos los valores por encima o por debajo del umbral, por lo que no se puede aplicar el método de backpropagation, dado que la principal herramienta de la que se sirve este para modificar los pesos son las derivadas (Bosch *et al.*, 2019).

- ***Función sigmoide o logística***:

$$\phi(x) = \frac{1}{1 + e^{\frac{-x}{\rho}}} \tag{C.2}$$

  Esta función cuenta con un parámetro ρ, que determina la forma de la curva. Esta función sigue dando una salida acotada entre cero y 1, no obstante, ahora contamos con una curva continua y diferenciable sobre la que podemos aplicar el backpropagation. Se utiliza en salidas binarias al igual que la función escalón, sin embargo, su salida esta vez puede interpretarse como una distribución probabilística binaria (Bosch *et al.*, 2019).

  El principal problema de la función sigmoide es que para valores extremos la derivada es muy pequeña y por lo tanto nuestra red aprenderá muy despacio, pudiendo llevar a lo que se conoce como la desaparición del gradiente (Sección 3.1.3).

- ***Tangente hiperbólica***:

$$\phi(x) = tanh(x) = \frac{senh(x)}{cosh(X)} = \frac{1 - e^{-2x}}{1 + e^{-2x}} \tag{C.3}$$

La función tangente hiperbólica es muy similar a la función sigmoide. La principal diferencia entre ambas es que el valor de salida de la tangente hiperbólica se acota entre -1 y 1. Los principales beneficios son que cuenta con una derivada mayor, por lo que proporciona un aprendizaje más rápido. Además está centrada en el cero, por lo que nos proporciona variación de signo entre diferentes estados de la red. Esto nos proporciona una mejora en el algoritmo del descenso del gradiente. Al igual que la sigmoide, su derivada prácticamente se anula para valores extremos (Bosch *et al.*, 2019).

- ***La función rectificadora (ReLU)***:

$$\phi(x) = max(0, x) \tag{C.4}$$

"Rectified Linear Unit", soluciona un problema no comentado previamente que es el coste computacional. Para la sigmoide y la tangente hiperbólica es muy elevado ya que cuentan de una o varias exponenciales, así pues, en términos computacionales ReLu es mucho más eficiente. Cuenta con un gran gradiente y no está acotada, por lo que soluciona el problema de la desaparición del gradiente, por lo que ha permitido el entrenamiento de redes más profundas (Bosch *et al.*, 2019).

El principal inconveniente es que puede crear las denominadas como "neuronas muertas", las cuales son llamadas así por que una vez que una neurona entra en valores de activación negativos no puede salir de ahí ya que su salida se vuelve nula, dejando así de participar en la red.

Se han propuesto diferentes funciones para solucionar este problema como Leaky-ReLu, P-ReLu, Gelu o Softplus (Xu, 2015). Todas ellas cuentan con ligeras desventajas que hacen que hoy en día ReLu siga siendo la función de activación principalmente utilizada. Entre ellas la más destacable es Gelu, que no es monótona, lo que aproxima más su comportamiento a las neuronas reales.

- ***Mish***:

$$\phi(x) = xtanh(\frac{1}{\beta}log(1 + e^{\beta x})) \tag{C.5}$$

Su predecesora se llama Swish ($y(x) = x\frac{1}{1+e^{-\beta x}}$)y al igual que ella (y GeLu) no es monótona. Cuenta con un mínimo lo que proporciona una activación para ciertos valores negativos evitando así las neuronas muertas. Computacionalmente es un poco más cara que ReLu. Es la que mejor resultados ha dado en visión computacional (Misra, 2019).

- ***Softmax***:

$$\phi^L{}_j = \frac{e^{z}{}^L{}_k}{\sum_k e^{z}{}^L{}_k} \tag{C.6}$$

Se utiliza como función de activación en las capas de salida de redes por dos de sus propiedades: los valores de salida son siempre positivos y nos da una distribución de probabilidad como salida de la red, ya que la suma de todos los valores de salida es 1. En muchas redes conviene interpretar (a_j) como la estimación de la probabilidad de que la salida sea correcta (Bosch *et al.*, 2019).